ISSN 0105-8517

Proceedings of the 12th IRIS – Part I

*I*nformation systems *R*esearch seminar *I*n *S*candinavia

August 13-16, 1989, Skagen, Denmark

Susanne Bødker, ed.

DAIMI PB – 296-I
December 1989

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

PREFACE

The IRIS seminars are small working conferences, where all participants contribute, mostly with papers, but also with posters, demonstrations, etc. The focus of the 12th IRIS was Creativity and Systems Development. To emphasize creativity, the conference was centred around workshops and other creative activities. Participants were asked to contribute with empirical experiences or theoretical contributions concerning creativity, or with suggestions for workshops, etc. to be conducted at the conference. As the conference site we had chosen Skagen, an old city at the northern point of Denmark, where, at the turn of the century, Scandinavian artists used to gather in the summer. We hoped that this once so creative environment, would be an appropriate environment for our creativity as well.

The program of the conference looked like the following:
Sunday
Dinner, presentation of participants, film on Skagen and creativity.
Monday
Introduction, group discussions of papers
Parallel work shops: STEPS (C. Floyd et al.), Role Playing (A. Kjær, M. Bartholdy), Future work shop (M. Kyng, S. Bødker).
Tuesday
Parallel work shops: An Artificial World (B. Dahlbom, L.E. Janlert), Prototyping (P. Kuvaja, K. Grønbæk), Playing DTP (D. Sjögren et al.)
Lectures in two parallel sessions:
• A (Chair: Lars Mathiassen): Erik Stolterman, Seppo Visala, Berit Thaysen/A. Munk-Madsen.
• B (Chair: Joan Greenbaum): Astrid Marie Reksnes et al., Niels Jacobsen, Marikka Pihlaja.
Hearing on design and creativity, chair: Pelle Ehn.
"Forum Theatre" on systems development.
Wednesday
Parallel work shops: Soft Systems Methodology (L. Mathiassen, P.A. Nielsen), HIS and Simulation (M. Nurminen et al.), Communication and assertiveness (R. Markussen).
Conclusion.

Organizers of the conference were Pelle Ehn, Lars Mathiassen and Susanne Bødker. The organizers would like to thank all the participants for some wonderful days. Grants from Nordiske Forskerkurser, Aarhus Universitets Forskningsfond, Statens Humanistiske and Statens Samfundsvidenskabelige Forskningsråd have made the conference a reality. Kirsten Kaadt, Karen Møller, Tove Legaard, and the rest of the administrative staff at the Computer Science Department, Aarhus University have been very helpful with respect to the many practical tasks surrounding the conference.

These volumes contain the papers presented at the conference, in lecture sessions and in working groups. Furthermore, some, but not all, of the workshops are reported on. The participants were:
Karen Albertsen, Liam Bannon, Jørgen Bansler, Merete Bartholdy, Ole Bisgaard, Gro Bjerkness, Sussie Brandrup, Tone Bratteteig, Bengt Brattgård, Lars Bækgaard, Keld Bødker, Susanne Bødker, Kathy Carter, Bo Dahlbom, Pelle Ehn, Inger Eriksson, Annika Finnäs, Per Flensburg, Christiane Floyd, Joan Greenbaum, Guido Gryczan, Kaj Grønbæk, Ann Hägerfors, Erling Havn, Ari Heiskanen, Austin Henderson, Jesper Holck, Kjell Åke Holmberg, Stig Holmberg, Niels Jakobsen, Lars Erik Janlert, Yana Kane-Essig, Karl Kautz, Pentti Kerola, Arne Kjær, Kari Kuutti, Pasi Kuvaja, Morten Kyng, Jens Kaasbøl, Birthe Lund, Kim Halskov Madsen, Randi Markussen, Lars Mathiassen, Preben Mogensen, Andreas Munk-Madsen, Bengt Möllerud, Peter A. Nielsen, Kenneth Nilsson, Markku Nurminen, Mads Nørby, Agneta Olerup, Birgitte Ravn Olesen, Marikka Pihlaja, Fanny Michaela Reisin, Astrid-Marie Reksnes, Timo Saarinen, Markku Sääksjärvi, Dan Sjögren, Jan Stage, Erik Stolterman, Lucy Suchman, Carsten Sørensen, Berit Thaysen, Michael Thomsen, Kari Thoresen, Randy Trigg, Timo Vara, Seppo Visala, Anders Wiberg, Daniela Wegge, Eleanor Wynn, Leikny Øgrim and Kristin Aardal.

Contents

DISCOVERING THE HUMAN ACTORS IN HUMAN FACTORS

Liam J. Bannon
Dept. of Computer Science
Aarhus University
Denmark

1 Introduction: A Question of Perspective

This essay discusses some problems and prospects for the field of human factors or ergonomics, specifically the more recent, and diversified field of human-computer interaction. Its main aim is to develop awareness of how an often unarticulated, though dominant perspective in the field can blind us to other more fruitful conceptions of human-computer interactions, and to emphasize the importance of shifting perspectives in the design process.

1.1 What's in a Name?

The terms we use to describe the objects of interest in a domain can be seen as reflecting our underlying conceptions about the domain. Naming things, as well as being a primordial human activity, has often had important social consequences (eg, being called a "witch" in the Middle Ages had rather unpleasant consequences for the person so named!). Through naming we can make the unfamiliar familiar, and vice versa, we can categorize, and reflect and act on the basis of these named objects. Once accepted, these terms can become a barrier to the reality that lies outside. For this reason, it is a useful exercise from time to time to re-examine the language we use to express our understanding of the world. Some of the concepts in the field of HCI are worth examining in this context, albeit briefly.

1.2 From Human Factors to Human Actors

I have chosen to highlight the human *factors* -human *actors* distinction in the title of this essay as it emphasizes the fundamental shift in perspective that can be brought about through naming. In other words, I claim that traditional *human factors* work, although it undoubtedly has merit, and has produced many improvements to existing technological systems, is often too limited in scope with respect to its view of the person. Within this approach, the human is often reduced to being another system component, with certain characteristics, such as limited attention span, faulty memory, etc. that need to be factored into the design equation for the overall human - machine system. This form of piecemeal analysis of the person as a set of components de-emphasizes important issues in work design such as human motivation and personality characteristics. People are more than a sum of parts, be they information-processing subsystems or physiological systems,they have a set of values and beliefs about life and work. It is the case that some researchers include such issues as work motivation and the desire for meaningful work under the topic of human factors, but this more encompassing interpretation of the human factors field is not signalled by the name, nor the bulk of studies appearing under this topic in the journals so designated. By shifting the term to "human actors" stress is placed on the person as an autonomous agent that has the capacity to regulate and coordinate his or her behaviour, rather than simply being a passive element in a human-machine system. The shift in terminology produces a shift in our perspective, emphasizing the wholistic nature of the person acting in a setting, in contradistinction to the view of the person as a set of information processing mechanisms that can be analysed in conjunction with the information processing mechanisms of the technology. (Related concerns about the role of human factors can be found in the work of Bjørn-Andersen, in a paper entitled "Are Human factors really human?" (1984). He also mentions this human factors/ human actors distinction, which I develop here, in Bjørn-Andersen (1986)).

1.3 The Concept of "Users"

Continuing our brief review of terms in the field and their hidden assumptions, take the concept of "users" that is ubiquitous in articles about the field. This general term refers to all people who use a particular computer system, or application. It can be distinguished from the term "operators" in that the latter implies a greater involvement with the machine or system, presumably one where the person is more uniquely assigned to the device. From the system design or human - computer interaction (HCI) research point of view, the term "user" can be seen as very egocentric. Our focus is on the technology, and our view of people is often simply as "users" of this piece of technology, and "naive users" at that! This can lead to problems. People may not know the technology, but they are *not* "naive" as to their *work*, rather it is the system designers that are "work naive"! There is nothing inherently wrong in taking this stance - talking of "naive users", though I prefer the less judgmental terms "casual" or "discretionary users" - for periods when designing computer applications, but there is a danger in thinking of people as *nothing but* "users". This can blind us to the fact that the "users" view of the technology we are developing may be very different to that of the designer's. From another perspective, the user is often a worker that has a set of tasks to perform, and their use of the computer may only be one element necessary to the accomplishment of their work. Neglecting this can lead to unworkable systems, due to the fact that necessary interactions between people and devices in the workplace may not be supported on the system, since the system designer never viewed the use of the system from this perspective, but persisted in having a machine-centered view of the overall human-machine system. It is the ability to shift perspectives, to be able see a problem from more than one viewpoint, to be able to empathize with, and understand the viewpoint of the users of the intended system that mark the good design team.

To try and encourage this shift in perspective away from a machine-centered to a user-centered view, we can witness the explicit usage of such "user-centered" terminology (see e.g. the title and contents of Norman & Draper (1986)). I just wish to flag this issue here to note that although talking of a "user-centered" approach to system design is an improvement, perhaps reflecting on the term "user" itself at times, as we do above, might be rewarding, and help us take a more appropriate view of users as people with work tasks and relationships which need to be also taken into account in the design of systems, especially those computer systems that are designed to support more general office functions, for example.

1.4 Active Users

While focussing attention on the user may be a positive step, we also must note that users (we will use this term frequently in the paper, though with the above caveat in mind) are not passive organisms which we must study and design for, but that they are *active agents,* wishing to accomplish tasks, to understand what is going on, and willing to jump ahead and explore on their own if the material is unclear or too pedantic. If the system does not give an explanation for its behaviour, the user will often try and make one up, in order to render the doings of the system comprehensible. People are always struggling to make sense of their world. Developing instruction sequences for users that are to be followed by rote, with inadequate explanation, fail to satisfy, or to work in most real situations. By way of illustration, here are the words of an office worker I interviewed that capture her wish to understand, to learn what's going on, that we should be aware of, and design for, in system design .

"People have to know how to understand, they have to be able to rationalize things out, to work things out, in a logical sequence. If they don't understand something, or how it works, then they have to go back to either just learning it by rote, or by asking, or by just making the mistake, and going back, and asking, and then correcting. But if you understand the system a little bit then sometimes you can think it out ....and if you can reason it out then it stays with you longer - it's easier to understand, to work with. But if you are just learning piecemeal, then you can't. If you are just learning by rote.....someone tells you, this is the way to do it,then you can memorize it, but you'll never fully understand it, and [never] be able to expand from there."(Direct quote from an interview I did at UCSD, 1983, with an office worker. See Brown (1986) for further argumentation on this point).

This is one area that has been the subject of investigation, with Carroll and his colleagues at IBM especially prominent (see Carroll and Rosson, 1987, for a synopsis of this work). Faced with this information, we should be careful about how we think about the capabilities of users. Just because thay do not understand how the machine works, or have difficulty with the system designer's terminology does not imply that they are stupid. The idea that we must design system' s so that "any idiot can use them" needs to be subjected to close scrutiny, quite apart from any moral reservations one may have about viewing people as idiots. Taking this as a serious design goal can often result in systems that *necessarily produce* such stupid behavior (see Bannon, 1986a, for further comment). Again, it is suggested that the system designer start out with the belief that workers/ users are competent practitioners with whom the system designer must collaborate with, in order to develop an appropriate computer system.

2 The Field of Human Factors and Human-Computer Interaction

2.1 Origins

The field of human factors is ostensibly concerned with building machines and artifacts in general in a way that fits more appropriately to human capabilities. While this goal is a laudable one, the particular context in which the field arose and developed is interesting. One can really trace a central strand in the human factors movement back to the work of the American engineer Frederick Winslow Taylor at the turn of the century, whose concern about the "one best way" of doing things involved him in very detailed analysis of manual worker's tasks and the tools they used to accomplish those tasks. From such studies, Taylor devised new artifacts for particular use situations, and determined the exact motor movements that the worker should perform to be efficient and maximize output. He was concerned about rationalising production, and concentrating knowledge of the work process in the hands of management, who could then instruct workers in the "correct" and most efficent means of carrying it out. Ideally, the workman would not be required to think about his actions - the separation of the conception of the work from its execution would be complete. To emphasize this, Taylor admitted that his ideal workman (for the manual tasks he had been studying at the Bethlehem Steel Foundry) would be "an ox"! Although his approach was called "Scientific Management", his interest was very much geared to the needs of managers and maximizing througput, rather than with scientific exploration *per se*, and certainly not with humanizing the workplace. Indeed, the whole concept of Taylorism became a *cause celebre* between labour and management due to the degrading way in which the "human element" was treated , with little emphasis on what motivated people to work other than straight economic gain. The later work of Frank and Lillian Gilbreth, who performed more scientific time and motion studies on workers in various occupations — and are more openly accepted as founders of the human factors movement — was not as closely connected with the needs of corporate business, but more research oriented, (see Giedion, 1948, for a brief account of their work) although many of their findings were useful to management.

So we see that from the very beginning the human factors movement has had a somewhat ambiguous connection with the improvement of the human condition, although it certainly contributed to more efficient human-machine systems. Much of the work was done at the behest of management and used to constrain worker freedom and autonomy. People were fitted to the machine, rather than vice versa. The same story was true in the area of selection of personnel for various jobs, where testing was performed by industrial psychologists to weed out those deemed unsuitable. Human values, the idea of meaningful work, did not enter into these schemas for some considerable time, and are missing in much of the classical-type work to this day. It appears that much of the basis of design in our society proceeds with a very strange mix of ideologies concerning work design, one part of which is concerned with de-skilling work and reducing labour to more peripheral and elemental processes, and the other that seeks to improve the quality of working life for these same people! This is a topic that has been commented on very succinctly by the engineer and scientist Howard Rosenbrock (1981). He deplores the gross misalignment between human abilities and the demands of many jobs and argues for the development of more "human-centered" systems, systems that allow workers to develop and build their skills in conjunction with the technology, and not systems that make these earlier skills redundant, with often a consequent loss to the world of such skills after a generation.

2.2 From Human Factors to Human -Computer Interaction

In the early days focus was more on getting machine systems to do something useful, and the concern with how easy it might be for someone to learn how to operate the system to do it, or invoke the required functions of whatever form, was not considered a high priority. People could be trained to perform whatever operations were required. As computing developed, the same was true, as those using the device spent years learning an arcane language to communicate with the system. This began to change as more people from disciplines outside of computing realised that the computer was just a symbol processing device that could be used on many different kinds of projects, not simply mathematical or statistical analyses. So the user community changed from one that was focused intrinsically on studying the properties of the machines themselves to discretionary users, people who saw themselves as having a job or profession that was not primarily geared to the computing medium itself, but wished to use it as a useful tool. This shift was encouraged by the spread of personal computers into the commercial world, making computational resources available to a much wider variety of people than heretofore.

Faced with this expanded user base, companies realised that the interface to their systems could make a huge difference to their attractiveness, indeed determining if the system would be used at all. It was in this context that the field of human-computer interaction (HCI) developed, encompassing a broader range of people and disciplines than the older, more traditional ergonomics or human factors type groups which had existed earlier in some settings, mainly working within military high-performance environments e.g. fighter aircraft cockpit displays, or in general occupational heath and safety environments, e.g. determining noise safety levels. Major influences in this expansion were cognitive psychologists, who realised that making a good computer interface was a matter not simply of physical, but increasingly, of cognitive ergonomics, and of software engineers, who were experimenting with the design of highly interactive interfaces, becoming concerned about how to conduct dialogues with users and how to present complex information to users effectively.

Over the last decade the area of human-computer interaction has grown enormously, both within academic research environments and corporate research laboratories. Despite the widespread interest, there is no clear set of principles that has emerged from this work. The experience of certain designers has been loosely codified, various long lists of design guidelines are available, and a large number of evaluations of existing systems have been produced, but the attempt to place this applied science on a more rigorous footing have not born fruit. Cognitive psychologists have studied particular issues, and we now know a lot about how people learn to use word processors, about the kinds of errors they make on different systems, about the mental models they attempt to construct of systems, but the application of such findings to new situations is not obvious. From the perspective of the designer, the work to date can highlight some pertinent issues, but in the search for new ways of thinking about and developing systems, there is not a lot that the field can offer.

3 Beyond current conceptions of HCI

Despite the legitimate advances that have been made in various arenas of human computer interaction (see Shneiderman (1987) for a collation of some of this material), there has been serious criticism of the field for its lack of relevance to practitioners in the systems design field. Commenting on a recent collection of papers (Carroll, 1987) that purported to sample current theoretical contributions to the HCI area, the reviewers' (Gray & Atwood, 1988) noted the lack of any examples of developed systems in the papers and the general lack of contact of the work with "real world" design situations. They explicitly state that the skeptical designer will not be convinced of the relevance of the cognitive sciences to the design of better human-computer systems based on this work. Within this collection, a discussion section by Whiteside and Wixon (1987) makes a number of pointed observations about the limitations of cognitive theory in its application to everyday design situations. There are a number of limitations in much of the current thinking about HCI that need to be remedied in order for the field to be more useful to designers in practical situations. I believe that the field must be seen from other perspectives, if it is to become more useful in the design and use of computer systems. Specifically, I believe we need to shift our focus somewhat, in the following directions:

3.1 From the Individual to the Group

The majority of HCI studies to date take as their focus the individual user working on a computer system. This is adequate for certain purposes, yet the uncritical acceptance of this situation as the norm in the field has meant that technical support for the ongoing conversations and work-related activities that span groups of people in real work situations have often not been handled properly. Workers often have difficulty in coordinating their activities through the computer system. The system then becomes a barrier rather than a facilitator for the co-ordination of work. Extending the focus of concern from the human-computer dyad to larger groups of people and machines engaged in collaborative tasks has emerged as an important area for research in the next period. The quick growth of this field, labelled Computer Support for Cooperative Work (CSCW), attests to its importance (see Greif, 1988, for a selection of papers in this area, and Bannon & Schmidt (1989) for an overview of the field).

3.2 From the Lab to the Field

Much of the early research done in the HCI field was confined to rather small controlled experiments, with the presumption that the findings could be generalised to other settings. Examples of such studies were those done on command naming conventions (see Barnard & Grudin, 1988 for a review of this research). It has become increasingly apparent that such studies suffer from a variety of problems that limit their usefulness in any practical setting. Firstly, by the time these studies are done the technology may make the original concerns outdated. Also important contextual cues for the accomplishment of tasks were often omitted in this transfer from the real world to the laboratory, and so the results of the lab studies became difficult to apply elsewhere. Increasingly, attention is shifting to *in situ* studies, in an effort to "hold in" the complexity of the real world situations, and a variety of observational techniques are being employed to capture activities, especially video.

The need to go outside the lab has been admitted by many HCI researchers in the past few years, including cognitive psychologists with a tradition of more rigorous laboratory studies (see Landauer, 1987). For example, Thomas and Kellogg (1989) discuss the ecological "gaps" that exist between the world of the research lab and the real world. We can see an increasing focus on the concept of "usability" among the research community — whether people can and do actually *use* the resulting systems designed for them (For an excellent tutorial on designing for usability see the chapter by Gould in Helander (1988). The Chapter by Whiteside, Bennett, and Holtzblatt on usability engineering in the same volume is also worth studying.) What these articles note is how difficult it is to evaluate the usability of an artifact without investigating the situations of use of that artifact (See Bannon & Bødker (1989) for an extension of this argument). From a design perspective, this means that we need to have a prototype or test system for users to experience in order to get information on the usability of the resulting system.

3.3 From Analysis to Design

Early human factors work tended to focus on evaluation of existing systems, and analysis of features that had been found in the use situation to be good or bad. However the concern of people in HCI now is how to build better artifacts, so we don't just want to know about systems after they have been built, we want to know how we should build them in the first place, and even what we should build. HCI should be a design science....."design is where the action is" — to quote a memorable phrase of Allen Newell. So the question is how can HCI contribute to the design of more usable and hopefully useful artifacts? One approach in HCI to this problem is represented in the work of Newell and Card (1985) who argue for the importance of approximate *calculational models* of the person performing a task that can be useful in the design process. The argument about the practicality and utility of such calculational models in general, and especially the claim that this is the the most important, if not the only way in which psychology and cognitive science can contribute to design, has been rather exhaustively discussed (Newell& Card, 1985, Carroll&Campbell, 1986, Newell & Card, 1986) and we will not re-hash it here, other than to voice support for a "science" of HCI that is broader than that conceived in the path-breaking, but limited work of Card, Moran, & Newell (1983).

Many groups are currently active in "user modelling", looking at the structure, content and dynamics of user cognition at the interface. Much work in the area continues the GOMS model tradition of Card, Moran, and Newell (1983), extending it in various ways. While meeting with some success in very narrow domains, there are acknowledged to be a number of rather serious problems in trying to extend the technique. The question is whether these problems are ones that can be overcome, or whether they are fundamental barriers to the use of such an approach in actual design situations. Our view tends towards the latter, as these are ideal models, of what users should do, not what they actually do, and they cover a very narrow range of user activities.

In this regard, the recent wave of interest in Programmable User Models (PUMs) (Young, Green & Simon, 1989) which are based on a generalized architecture of human cognition seem to be also unduly narrow. Rather than moving designers closer to actual users, such a device, if it existed, would seem to support the view that real contact with users was unnecessary, as the designer could just program the PUM in order to understand the "human constraints"! The very vision of a PUM seems to us a rather abstract view of human activity in the world, and to imply a rather strange relationship between designers and users. As Reisner (1987) notes in her discussion of earlier modelling work, such work can never replace prototyping and actual empirical user testing, although it might have a role at a certain stage in the design of a new system.

3.4 From User-Centered to User-Involved Design

As noted earlier, the term "User Centered System Design" has been used for some time, in order to focus attention on the needs of the user rather than on the system hardware and software possibilities. Yet what this phrase really means, or how it can be achieved, is far from clear. In some cases it dissolves into platitudes such as "Know the User". Such kinds of general guidelines are of little use in practical situations of design, due to their lack of specificity. A more radical departure from much current thinking within the mainstream HCI world is to look on users not simply as objects of study, but as active agents within the design process itself. This involvement of users in design is seen not only as a means for promoting democratization in the organizational change process, but also as a key step to ensure that the resulting computer system adequately meets the needs of the users.

Adequate input from users has been presumed to have been captured in the requirements analysis phase of the project, and task analyses done earlier by the system team. Over the years, it has been acknowledged that these are often inadequate, and the question has begun to be asked whether this is because of some problems in the way of doing the studies locally, or whether there is a fundamental problem with the very assumption that we can map out users needs and requirements successfully in advance through simple techniques of observation and interviewing. Many now argue that users need to have the experience of being in the future use situation, or at least an approximation of it, in order to be able to give comments as to the advantages and disadvantages of the proposed system. So, some form of mock-up or prototype, needs to be built in order to let users know what the future use situation might be like.

4 Conclusion: HCI in Systems Design

The need to understand the user's needs, and to understand the tasks performed by the user is basic to the system development process. However, it is a mistake to think that simply having a human factors person on the design team is by itself sufficient to ensure that the "human factor" has been adequately taken into account, in the sense that is being discussed here. Even in companies where there exist groups specifically targeted to give "human factors" advice on projects, one often finds that they have little influence over the design process, often regarded as "add-ons" by the engineering staff. This state of affairs has been unfortunately encouraged by the human factors personnel themselves, who often seem unwilling to really understand the complete project, or product, but focus on the narrow aspects that are adjudged to require human factors input.

Human factors, or ergonomics considerations tended to be incorporated into the design process simply as a set of specifications that the delivered system must adhere to. The actual work of the human factors personnel consisted of operator task analyses to be fed into these specifications and perhaps some interface re-touching near the end of the development cycle, when the system design had already been fixed. In general the role of these personnel was seen as ancillary to the main task of building the system.

What is being advocated here is an approach that, while acknowledging the contribution that different disciplines can make to the design process, ultimately depends upon the users themselves to articulate their requirements, with the system design team, composed of a variety of specialists, acting in the capacity of consultants to the project. Designers and users must be prepared to acknowledge each others competencies and to realize that effort must be made by both parties to develop a mutually agreed upon vocabulary of concepts that can be shared across the different groups that comprise the project. It is no easy task for different disciplines and work activities to accomplish this, and it is in this area that additional research would be valuable.

Some of the efforts in Scandinavia (see for example the papers in Bjerknes et al, 1987) on involving users in design provide a promising start towards the alternative systems design paradigm advocated here. Within such an approach, the starting point, and the end point of the design process is with the users themselves, from what they require, to how they evaluate the prototype, and the iterations that follow. Along the way, the services of a variety of disciplines may be required, not just the software engineer and the ergonomicist, but perhaps also architects, sociologists and anthropologists. These disciplines should come together in the overall design process as required, and not dictated by some arbitrary flow model by which the system design gets handed around sequentially from one discipline to another. It is in the mutual interaction of these different perspectives, focused on a particular design project, that good design may emanate.

References

Baecker, R.M. & Buxton, W. (Eds.) (1987) Readings in Human-Computer Interaction: A Multidisciplinary Approach. Los Altos, CA: Morgan Kaufmann, 1987, pp.41-54.

Bannon, L. (1986a) Helping Users Help Each Other. In Norman, D. & Draper, S. (Eds.) User Centered System Design. London: Erlbaum, 1986.

Bannon, L. (1986b) Computer-Mediated Communication. In Norman, D. & Draper, S. (Eds.) User Centered System Design. London: Erlbaum, 1986.

Bannon, L. & Bødker, S. (1989) Beyond the Interface: Encountering Artifacts in Use. Dept. of Computer Science, Aarhus University, DAIMI PB-288, Oct. 1989.

Bannon, L. & Schmidt, K. (1989) CSCW: Four Characters in Search of a Context. In Proceedings of First European Conference on Computer Support for Cooperative Work, London.

Barnard, P. & Grudin, J. (1988) Command Names. In Helander, M. (Ed.) Handbook of Human-Computer Interaction. (Amsterdam: North-Holland, 1988).

Bjerknes, G., Ehn, P. & Kyng, M. (1987) Computers and Democracy: A Scandinavian Challenge. Aldershot, UK: Avebury.

Bjørn-Andersen, N. (1985) Are "human factors" human? In INFOTECH State of the Art Report 13: 1, Man-Machine Integration, 1985.

Bjørn-Andersen, N. (1986) Understanding the nature of the office for the design of third wave office systems. Paper presented at Workshop on Modelling the nature of the office for design of third wave office systems, Scæffergården, Denmark, August 1986.

Brown, J.S. (1986) From Cognitive to Social Ergonomics and Beyond. In Norman, D. & Draper, S. (Eds.) User Centered System Design. London: Erlbaum, 1986.

Card, S., Moran, T. & Newell, A. (1983) The Psychology of Human-Computer Interaction. New Jersey: Erlbaum, 1983.

Carroll, J.M. (Ed.) (1987) Interfacing Thought: Cognitive Aspects of Human-Computer Interaction. (Cambridge: Bradford/ MIT Press, 1987).

Carroll, J.M. & Campbell, R.L. (1986) Softening Up Hard Science: Reply to Newell and Card. Human Computer Interaction, 1986, Vol. 2, 227-249.

Carroll, J.M. & Rosson, M.B. (1987) The paradox of the active user. In Carroll, J.M. (Ed.) Interfacing Thought: Cognitive Aspects of Human-Computer Interaction. (Cambridge: Bradford/ MIT Press, 1987).

Giedion, S. (1969) Mechanization Takes Command. (New York: W.W. Norton, 1969)

Gray, W.D & Atwood, M.E. (1988) Review of Interfacing Thought: Cognitive Aspects of Human-Computer Interaction by Carroll, J.M. (Ed.) (Cambridge: Bradford/ MIT Press, 1987) in SIGCHI Bulletin, October 1988, 88-91.

Greif, I. (Ed.) (1988) Computer-Supported Cooperative Work: A Book of Readings. San Mateo, CA: Morgan Kaufmann, 1988.

Helander, M. (Ed.) Handbook of Human-Computer Interaction. (Amsterdam: North-Holland, 1988).

Landauer, T. (1987) Relations between cognitive psychology and computer systems design. In Carroll, J.M. (Ed.) Interfacing Thought: Cognitive Aspects of Human-Computer Interaction. (Cambridge: Bradford/ MIT Press, 1987).

Newell, A. & Card, S.K. (1985) The Prospects for Psychological Science in Human-Computer Interaction. Human Computer Interaction, Vol. 1, 209-242.

Newell, A. & Card, S.K. (1986) Straightening out softening up: Response to Carroll and Campbell. Human Computer Interaction, 2, 251-267.

Norman, D. & Draper, S. (Eds.) (1986) User Centered System Design: New Perspectives on Human - Computer Interaction. London: Erlbaum, 1986.

Reisner, P. (1987) Discussion:HCI - what it is and what research is needed. In Carroll, J.M. (Ed.) Interfacing Thought: Cognitive Aspects of Human-Computer Interaction. (Cambridge: Bradford/ MIT Press, 1987).

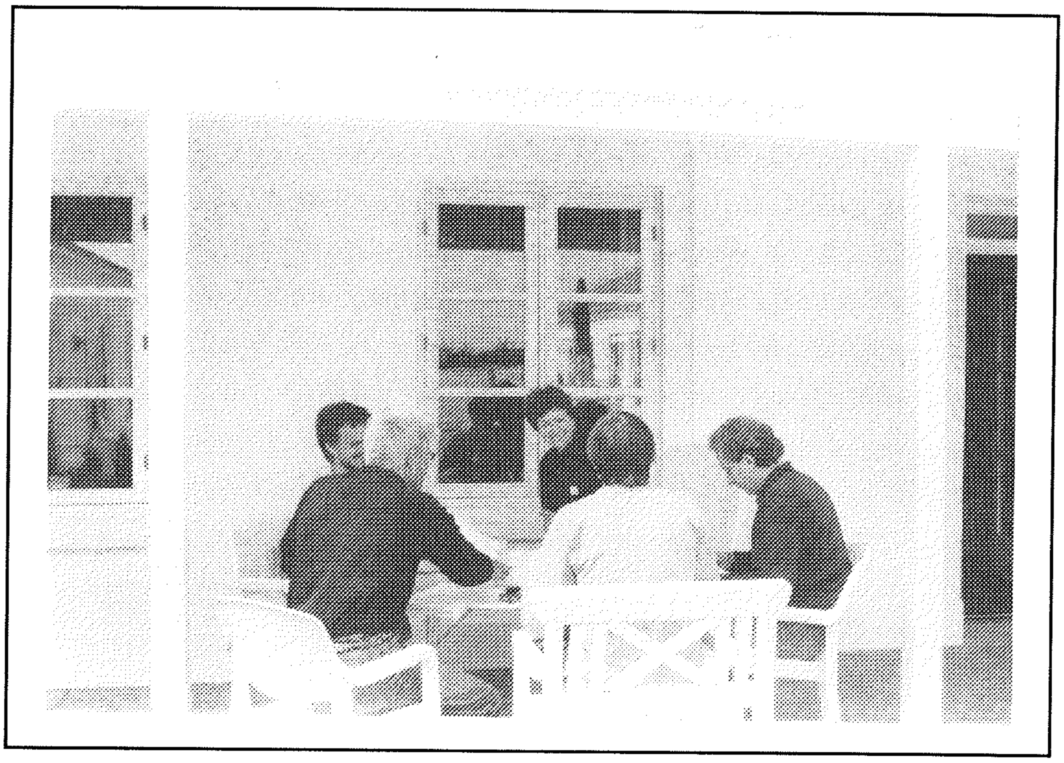
Rosenbrock, H. (1981) Engineers and the Work that people do. IEEE Control Systems Magazine, 1,3, 4-8, 1981. (Reprinted in Pylyshyn, Z. & Bannon, L. (Eds.) Perspectives on the Computer Revolution ,New Jersey: Ablex, 1989).

Shneiderman, B. (1987) Designing the User Interface: Strategies for Effective Human-Computer Interaction. Reading, Mass.: Addison-Wesley, 1987.

Thomas, J. & Kellogg, W. (1989) Minimizing Ecological Gaps in User Interface Design. IEEE Software, January, 78-86.

Whiteside, J. & Wixon, D. (1987) Improving human-computer interaction—a quest for cognitive science (Discussion). In Carroll, J.M. (Ed.) Interfacing Thought: Cognitive Aspects of Human-Computer Interaction. Cambridge: Bradford Press, 1987, pp. 337-352.

Young,R.M., Green, T. , & Simon, T. (1989) Programmable User Models for predictive evaluation of user interface designs. In Proceedngs of CHI '89 Conference on Human Factors in Computing Systems, New York: ACM Press.



Karl Kautz, Jesper Holck, Tone Bratteteig,
Anders Wiberg and Morten Kyng doing group work

COMPUTERS AND CONFLICTS

Merete Bartholdy and Arne Kjær
Institute of Information- and Media Science, Aarhus University, Denmark

Abstract: Computers and Conflicts is a game developed for use in teaching systems development. The game is a mixture of "chance" (throwing of dice) and role playing and it is inspired by an adventure game called Dungeons and Dragons.

Computers and Conflicts is intended to teach issues concerning the context of systems development, rather than tools and techniques for systems development. The students experience of the game is used as input to a discussion on systems development.

In this paper, we describe the purpose and origin of the game, an example of how to use it to teach systems development, and a discussion of the game with regard to its further development.

Computers and Conflicts - Purpose and Origin

Computers and Conflicts was inspired by Dungeons and Dragons[1]. That game is played by a Dungeon Master and some number of other players, each of whom controls a Role Character. A Role Character can be a warrior, a knight, a thief or the like. The Role Characters have joined together to look for treasures. In the game they have to find their way through a dungeon which only the Dungeon Master is familiar with beforehand. On their way, the Role Characters meet dragons and monsters. The players decide how their Role Character should behave but at times, for instance in a fight with a monster, dice are thrown to decide the outcome of the fight.

We have found this game interesting for several reasons, the most important of these are:

- The players act as a group - they do not fight each other.
- The game is a good way of showing that nobody is good at everything and that everybody is good at something. Some Role Characters are good at infighting, others with bow and arrow. And some are good at sneaking around. Through the game the players learn to use each others (the Role Characters) best abilities.
- The game can be played over a long period of time allowing the Role Characters to gain more experience; this way the characters change and develop through the game.

[1]*Dungeons and Dragons*, Gary Gygax, Dave Arneson, TSR,1972

- The game supports fantasy and illusion, i.e. for a short while the players seriously discuss dragons and monsters, and how to get through the dungeon.

It is these properties of Dungeons and Dragons we wish to use in teaching systems development.

The context of systems development

When learning systems development today students are taught programming, user interface design, prototyping, various systems development methods as well as other subjects.

When we teach any one of these subjects, certain issues are hidden. When we teach programming, the students get usually get a detailed specification, so the problem of deciding what tasks the programme is to execute is hidden. When we teach user interface design the problems of getting the right setting for this are hidden, and so on.

Some hidden issues of one subject are raised by other subjects. But, looking at the collection of subjects we have today, we feel there are still hidden issues. These are connected to the context of systems development.

Systems development is a process taking place in an organization with many different user groups, managers, several old computer systems and so on. Systems development is one process of many bringing about change in the organization. For instance, the overall purpose of the systems development process could easily be, not only to get a computer system, but also to introduce some organizational changes.

As the teacher, you can give lectures on project management, project evaluation, different interest groups, organizational problems and so on. But all you do is give the impression of different kinds of problems, most of the students lack a praxis to relate the points of the lectures to. They can accept or not accept your points, but they can not discuss them. Furthermore, your lectures may easily end up sounding prejudiced or pessimistic.

The fact that Dungeons and Dragons so well supports fantasy and illusion made us believe that transferring the game into teaching systems development could be a good way to introduce the context of the systems development process into education. Issues including:

- organizational problems
- political problems
- compromises
- intrigues
- time pressure

The playing of the game gives the students some experience and with that the ability to discuss issues like the above. We have no ambition of giving the students a realistic experience of real life systems development, but rather a starting point for a discussion.

The game is intended to be of many aspects of teaching systems development. As such it raises issues concerning the context into the teaching of systems development, but during a play of the game other issues will, of course, remain hidden. The game does not teach the students programming, user interface design, or the like.

Besides the introduction of more issues related to context, we expect the following achievements:

- Teaching the students to use each others best abilities.
- Teaching the students to organize a systems development process, to decide what techniques and tools to be used in a specific situation, and why.
- The game will introduce a way of teaching that promotes a greater students involvement.

Dungeons and Dragons

The origin of Computers and Conflicts is Dungeons and Dragons. To help the reader understand the design choices of Computers and Conflicts, we present a short introduction to Dungeons and Dragons here.

In Dungeons and Dragons the Role Characters are described by a Character Record Sheet, assigning numbers to abilities such as Strength, Intelligence, Wisdom, Dexterity and so on. Also, the Role Characters have a certain number of Hit Points, which can be lost incrementally in a fight (when a character reaches 0 in Hit Points, he or she is dead).

The Dungeon Master has similar Character Sheets for the monsters that the group of Role Characters meet. If one of the Role Characters decides to fight a monster the player throws a die twice. The outcome of the cast is judged according to the values of the abilities of the Role Character sheet. The first throw of the die decides wether the monster has been hit or not, if it is hit, the second throw of the die decides how much damage the monster has suffered. After this the Dungeon Master plays the monster fighting back the same way, in this case the second throw of the die decides how many Hit Points the Role Character loses.The Role Characters do not fight all the time, and not all of the "monsters" are hostile. When the group meets an unknown Character, they can decide to attack it, find a way to avoid it, or even attempt to persuade it to join the group.

A game of Dungeons and Dragons usually takes place over a long period of time, for instance a group of players may meet once a week for months or even years. The same group of Role

Characters can, if they survive, join several different adventures organized by the Dungeon Master. Through these adventures the Role Characters can develop - they become more skillful.

The Rules of the Game

In Computers and Conflicts you do not fight - at least not with physical weapons and not with the aim to kill.

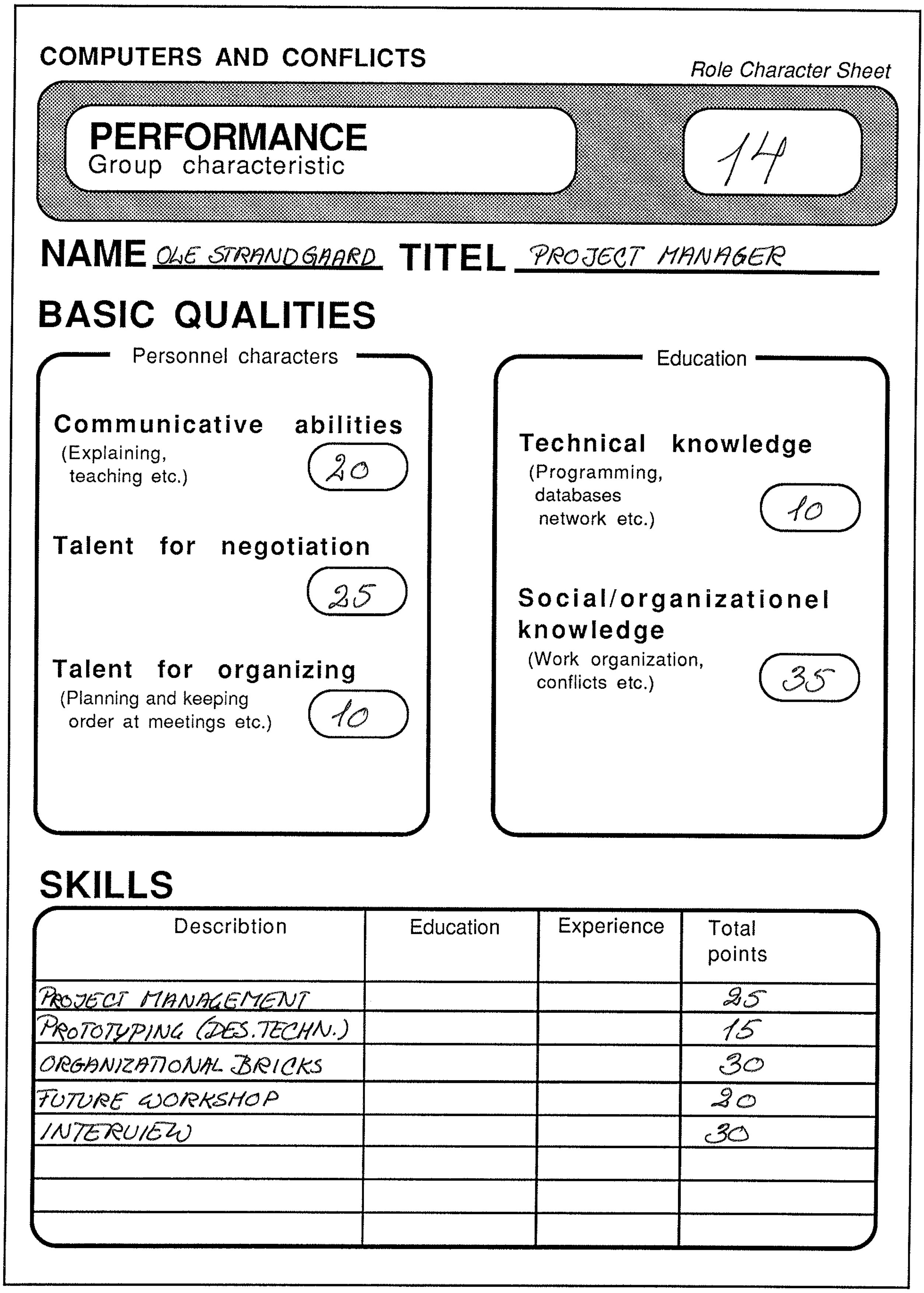
The Role Characters of Computers and Conflicts form a project group in charge of a systems development process. This may include developing and introducing of a new system, introducing some standard system, producing a requirements specification, choosing hardware configuration, and the like.The Characters are employed in a consulting firm, so at the beginning of the game, they know nothing about the user organization.

The "fights" of Computers and Conflicts are meetings, negotiations, interviews, analysis and other activities of the systems development process. Each time the project group decides to perform one of these activities, dice are thrown to decide the success of the activity. Such activities of systems development could include:

- A meeting with persons from the user organization
- A phone call to arrange a meeting
- Making a data flow analysis of the organization
- Making a prototype in collaboration with a group of users

Role playing has a much larger place in Computers and Conflicts than in Dungeons and Dragons. When playing the game we use two table - one, the project-group-table, where the dice throwing and the internal project group discussions take place, and the other, the theather-table, where scenes of specific actions like a meeting or a negotiation are played - the participants play the Role Characters and the Game Masters play the other Characters. For instance, if we look at the above list of activities, the phone call and the meeting are acted as stage plays (role plays), while the data flow analysis and the prototyping are handled like activities with a certain output; that is the players do not perform them they just get the results of the activities.

The Role Character of Computers and Conflicts are, like in Dungeons and Dragons, described by a number of values on a Role Character Sheet (fig.2). These values are used by the Game Masters to decide the probability of success or failure.



COMPUTERS AND CONFLICTS

Role Character Sheet

PERFORMANCE
Group characteristic

14

NAME OLE STRANDGAARD **TITEL** PROJECT MANAGER

BASIC QUALITIES

Personnel characters

Communicative abilities (Explaining, teaching etc.)	20
Talent for negotiation	25
Talent for organizing (Planning and keeping order at meetings etc.)	10

Education

Technical knowledge (Programming, databases network etc.)	10
Social/organizationel knowledge (Work organization, conflicts etc.)	35

SKILLS

Describtion	Education	Experience	Total points
PROJECT MANAGEMENT			25
PROTOTYPING (DES.TECHN.)			15
ORGANIZATIONAL BRICKS			30
FUTURE WORKSHOP			20
INTERVIEW			30

Fig.2: *A Role Character Sheet from Computers and Conflicts*

The probability is expressed by dividing the range from 1 to 100 into the following five intervals: Disaster, Failure, Reasonable, Success, and Extreme Success. At the beginning the range is as depicted in figure 1.

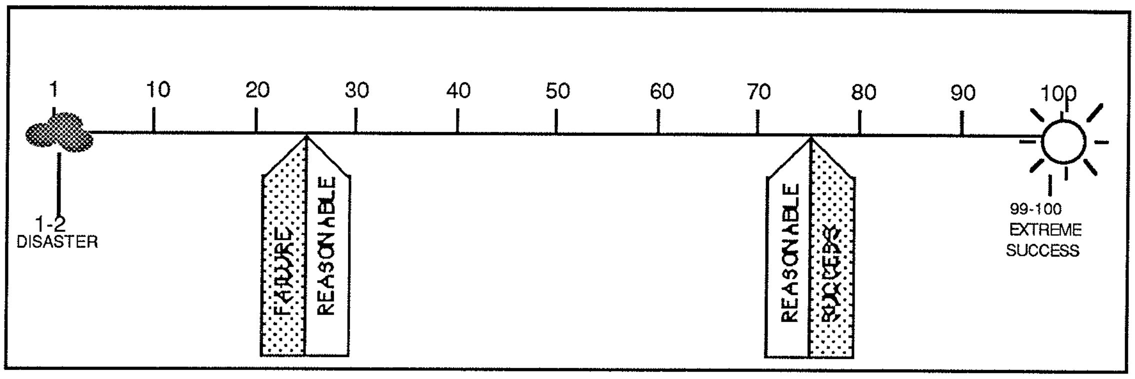


Fig.1: *The Probability ruler shows how the range from 1 to 100 is divided. It looks like this, before the calculation of the probability begins.*

In the following, the arrow that determines the border between Failure and Reasonable is called the *lower level*, and the arrow that determines the border between Reasonable and Success is called the *upper level.*

On the basis of the relevant values from the Role Character Sheet, the Game Masters calculate a specific probability. The Lower Level and the Upper Level can be moved by this calculation, but the limits for Disaster and Extreme Success remain unchanged. As a consequence no matter how good your chances are, you will always have a 2% risk of making a disaster of the activity in question. And no matter how poor your chances are, you will always have a 2% chance of making an Extreme Success.

The final outcome is decided by throwing two ten-sided dice - one for each digit. Throwing 00 counts for 100.

After the throw, the scene of the activity if "enactable" shifts to the theater-table - in the ensuing role play, the players play the Role Characters and the Game Masters play the rest of the Characters. The Game Masters decide how to play their roles, according to the throw. If the throw was a Failure, the Game Masters play antagonistically, and if the throw was a Success, the Game Masters play co-operatively.

To give some examples of this, let us look at the above list of activities. If the Project Group decides to have a meeting with employees from the user organization, they first have to select someone to phone and arrange the meeting.

Now, the Game Masters regulate the probability ruler according to the groups Performance and the relevant values of the phoning persons Role Character Sheet, in this case his or her communicative abilities. After this the dice are thrown, and finally the scene is played at the theater-table.

The Game Masters interpret what a Failure or Success means in this specific state of the game. If the throw of the dice leads to a failure, then the Game Master at the other end of the phone might say: "Sorry, I'm going to a conference, the meeting can, at the earliest, be in two weeks." If the throw of the dice was a success, the Game Masters act interested and co-operative, and the phone call leads to a timely arrangement of the meeting.

In the case of the interviews, the scene is not played at the theater-table. The Game Masters hold all the information on the organization and know what information could come out of the interviews. If the throw of the dice is a success, the Game Masters provide the Project Group a lot of this information. If it is a failure, the Game Masters provide little information, they may even intentionally mislead.

Note that, the concepts of success and failure are different from what might be expected. When the dice show a failure it does not preclude success - but only that the players get a hard time. Since the concepts of success and failure only affect how the environment acts, one can, through ingenuity and hard work, turn a failure into a reasonable activity or even a success. So success and failure are more like good conditions and bad conditions - like good times and hard times.

In the following a more detailed description of the values of the Role Character Sheet is given.

Performance

The intuitive interpretation of a groups Performance is, that it expresses how well the project group is respected. With a high Performance, the group has a better chance of making a success out of an activity. On the other hand, if the Performance gets low, the risk of Failure increases - that is to say, it does not matter how good your ideas are, because people just do not trust you anymore.

The Role Character in fig.2 is a member of a group with a Performance of 14, that is above average. At the beginning the groups Performance is the result of a throw of a die - you throw an eight-sided die and add seven. In the game the Performance value is used to regulate the probability ruler - if the Performance is above 12 the success interval will increase, if it is below 11 the failure interval will increase. The average Performance is 11-12, and this value will not affect the probability ruler.

Through the game, the Performance of the group can change - if they make a Success the Performance increases, if they make a Failure the Performance decreases.

In some sense, the concept of Performance compares to that of Hit Points in Dungeons and Dragons. If a group is very unfortunate and loses all its Performance, they get fired! - But you have to have very bad luck to reach that point.

Basic Qualities

The Basic Qualities convey what kind of person a Role Character is, and what kind of education he or she has. A Role Character with a high value in Technical knowledge and a low value in Social/organizational knowledge could for instance be an engineer. The Role Character in fig.2 could be a person educated at our institute (information- and media science). The values of the Basic Qualities are distributed by the players according to certain rules. Distributing the values corresponds to deciding what different persons should be in a project group. The numbers on the Role Character Sheet do not, of course, give a full description of a person. The players have to fill in the rest of the role with their own personal qualities. Since the players occasionally have to play their Role Character, it is advisable to distribute the Basic Quality values so they match the players own qualities.

In almost any activity the Characters participate in, one or more of the Basic Qualities are needed. The Game Masters and the players negotiate on which Basic Qualities are relevant to a specific activity. Like the Performance, the Basic Qualities of the Role Characters involved in the activity are used to regulate the probability ruler. The Value 25 is the average and has no affect on the probability. A higher value increases the success interval, and a smaller value increases the failure interval.

The values of the Basic Qualities are distributed by the players, according to the following rules:

- The total of all the values must not exceed 100.

  This gives everyone an equal chance. The participants are students at the same level, so it would not be fair to give some fewer points than others.

- No Basic Quality should be given more than 45 nor less than 5.

  Because of the way the ruler is made, the player will not benefit from points values higher than 45. The Basic Qualities are chosen so that everyone has a minimum number of points, thus the lower limit of 5.

The Basic Qualities can be compared to the Abilities on the Dungeons and Dragons Character Record Sheet.

Skills

Skills have values just like the Basic Qualities. But unlike the Basic Qualities, all the Role Characters do not have the same Skills. Skills refer to special tools or techniques, that you wish to use in the systems development process. The Players decide their own skills according to how they want to develop the system.

Flow-oriented analysis and description (Yourdon/DeMarco-like)
Prototyping
Programming in 3. generation language (f.i. PASCAL)
4. generations tools
Database design and administration
Data transmission and local area network
Future Workshop
Qualitative interviews
Questionnaires and statistical analysis of these
Analysis of consequences (user-oriented)
Ergonomics and analysis of work environment

Fig. 3: *Examples of skills players can choose*

The different skill values are again used to regulate the probability ruler. The Game Masters negotiate with the players about which skills are relevant to a specific activity, if any. For instance, doing a phone call or meeting may not require, unlike the activities of interviewing and prototyping.

The players have a certain number of Skill Points to distribute at the beginning. Again 25 is the average, and the players have to decide whether they want to be good at a few things or capable of doing a number of things, but none of them very well. If the value of a relevant skill is above 25 the "reasonably"-interval shifts downwards, making the success interval larger and the failure interval smaller. If the value is below 25 the "reasonably"-interval shifts upwards, making the failure interval larger and the success interval smaller.

The Role Characters can gain skills both through education and through experience. If for instance, during the game, a Role Character participates in arranging a future workshop, he or she gains a certain amount of experience points, implying that the next time this Character arranges a future workshop, the chances of success are higher. Also, the project group can send one or more members of the group to courses in using special kinds of tools or techniques, and thereby gain more education points. Together, education points and experience points form the total Skill points for one skill.

Time

Computers and Conflicts would be too unrealistic if time did not play a major role. Time can pass in two different ways in Computers and Conflicts, either *by jumps* or *continuously.* Whenever the players decide to perform an activity in the systems development process, it "costs" in time according to a "price list" that only the Game Masters have access to.

Time is measured in days (one person) and weeks (one person). As an example, suppose the five persons in the project group attend a one day meeting. This is counted as 5 days (one person), which is the same as 1 week (one person).

Counting time this way only would not include the time spent on planning and internal project group discussions. So, whenever the players are placed at the project-group-table - and they are not throwing dice, discussing with the Game Masters, or the like - a stop watch is running. Two minutes on this stop watch count as 1 day (one person). In other words, five minutes discussion around this table count as if the five members of the project group had had half a days meeting. When the players have finished their project group discussions, the minutes on the stop watch are transformed into days and weeks, and the stop watch is reset.

The days and weeks are counted on a flip-chart, providing the players information time spent this far.

An example

A game of Computers and Conflicts consists of three phases. The first phase we call *preparation.* Here the settings is presented say, through a short story on how the project got started is told, the participants design the Role Characters, and then prepare the first activities of the game. The second phase is *the actual play of the game*, and here the participants have to take the initiative on various activities and react to what other characters, played by the Game Masters, are doing. The third phase is *reflections* on the game. Reflections would include: Why did the participants act the way they did, what other possibilities did they have, how could the Game Masters Characters have responded differently, at what points was the game realistic/not realistic and why, and what may the participants learn about system development in such an game. Another kind of reflection, that is not part of the game, is about the intentions and design of the game, for instance the combination of the use of dice and actual role play, the concept of "performance" as a way of measuring success in projects, and so on. In this section we describe the three phases of one round of one game. In the next section some general reflections on the game are discussed.

Preparation

An introductory story

The history at Peter Hansen, Ltd.: Peter Hansen Ltd. is a manufacturing company, producing two types of related products. They want a new stock-control system. The company has two plants, each producing one of the product types.

At the moment, Peter Hansen Ltd. has a computerized stock-control system, but it is a very simple batch-oriented system. Hans Olsen, the sales manager at Peter Hansen Ltd., has initiated the project, and has permission from the board to make contact with a firm of consultants, Soft Data Inc, to get the system developed. There have been a couple of meetings between Hans Olsen and a senior consultant and the sales manager from Soft Data Inc. The result of these meetings has been an agreement on a contract.

The history at Soft Data, Inc.: The participants in the play form a project group at Soft Data Inc. Originally, the senior consultant should have been the project manager, but the firm got a more important job for her. So, the project group gets the job, knowing only what is written in the contract. Each member of the project group works 20% of a full time job on the project, with outside responsibilities in the remaining 80%. Soft Data has estimated 400 consultant hours for the first phase. To an activity like this, Soft Data assigns 25 h. as one persons job in one week. So, the project group can use 16 person-weeks in the first phase.

The first job of the project group is to make a detailed plan of activities and a detailed time schedule for the first phase. This plan is to be presented to a group manager at Soft Data Inc., Georg Sørensen, next week.

Designing the role characters: The participants first discuss what types of characters should be in the project group (basic qualifications), and what skills these characters should have. We have designed the game in this way, although we know that it is not realistic for the project group to design its own characters. But it gives the participants an opportunity to reflect both upon the requirements for basic qualifications and the different skills they find useful to have in the group. And it is, after all, a learning process for the participants.

Then the project group carries out the first activity, designing a project plan to present to there group manager in their own company. When teaching we inserted a time interval of two days between the introduction of the game and the actual playing (e.g. between the first and the second phase). The project group used this time to design the project group and the characters in it, and to make the project plan.

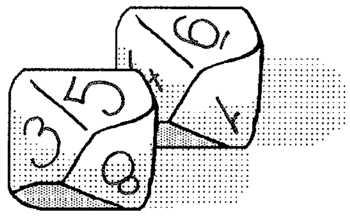
The game

Setting the scene: In a game like this, where fantasie and illusion are of great importance, the scene, the environment, should support this illusion, as well as serious participation in the game. Furthermore, only a few of the students can participate in the game, to the rest of the students, the audience, this should be a kind of theater. The audience is supposed to identify themselves with members of the project group. Setting the scene also requires that the players presents the Role Characters they have designed. In figure 4 you see how these look.

Name	analyst (project manager) Ole Strandgaard	system analyst John Jensen	system analyst Ulla Knapstrup	programmer Pia Tuesen	system programmer Jesper Smidt
Basic qualifications:					
Communicative abilities	20	20	30	30	10
Talent for negotiation	25	10	5	5	10
Talent for organizing	10	20	10	10	30
Technical knowledge	10	30	20	45	45
Social/organizational knowledge	35	20	35	10	5
Skills					
Project management	25				
DeMarco, Structured Analysis		30		10	10
Object oriented analysis (Jackson)			30	10	10
Prototyping as a design technic	15	15	25		
Prototyping as a programming technic				20	
4. generation tools		30		30	
Hyper-media			30		
Organizational bricks	30				
Role play			25		
Future Work Shop	20				
Interview	30				
Analysis of work environment		25			
Programming in high level language				30	40
Database design					40

Fig. 4. *Example of the composition of a project group with Basic Qualities and skills.*

After the scene is set, the actual play begins. First of all the Project Group has to throw an 8-sided die and add 7 to define the groups Performance. The dice shows 7, so the performance will be 14. They then have to decide who is going to the meeting with the group manager, Georg Sørensen. They select the project manager to handle this meeting.



But first, the dice have to be thrown, and before that, the Game Masters have to adjust the ruler. Firstly, the project manager has the skill of "project managing", but 25 is the average, so the ruler is unaffected. Secondly, the Game Masters decide, that the Basic Qualities needed are "Communicative abilities" and "Talent for negotiation". The average of these two is 22,5 for the project manager, so the lower level is adjusted to 27,5 causing the "Failure"-interval to increase. Finally, the performance is 14, which means that the upper level will be moved from 75% to 69%. Now the dice are thrown - the result is 56, and that is in the "reasonably"-interval.

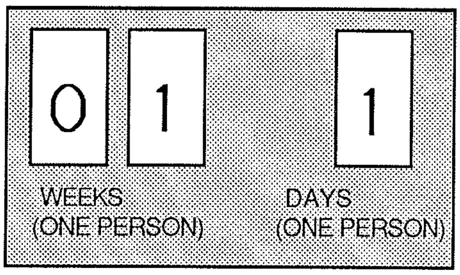
The project manager and one of the Game Masters now play the meeting as a role play. One Game Master plays the role of group manager Georg Sørensen at Soft Data Inc. In the role playing, the Game Master uses the number 56 to interpret, how to respond to the project plan presented by the project manager. The project plan is shown in figure 5.

1. Interviews of people in different parts of the organization	50 h.
2. Analysis of the present way of doing stock control, incl. data flow analysis and an organizational analysis using "bricks"	75 h.
3. Future Work Shop	75 h.
4. Prototyping as part of the design	50 h.
5. Project management and meetings in the project group	50 h.
6. Not disposed	100 h.
Total	400 h.

Fig. 5. *The proposed project plan*

Role play I:

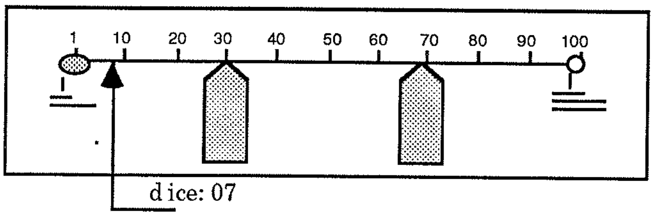
At the beginning of the role play, the project manager presents the plan. First of all Georg Sørensen is satisfied with the different activities presented in the plan. But he would like to have more details, for instance, specific deadlines for the activities. Further he believes that 50 h for project management is too little, but he thinks it is a good idea not to dispose all the hours at this time of the project. He asks what kind of data flow analysis they are planning to do. The project manager answers that it is an overall analysis of the present data flow in the organization related to stock control. The data flow analysis should serve the purpose of a learning process for the project group. After discussing some other details in the plan, the project manager asks who they are going to contact and make arrangements with at Peter Hansen Ltd. The group manager suggests, first of all, that they call the sales manager, Hans Olsen, who initiated the project. At the end of the meeting, the project manager promises to *come up with a more detailed plan when they know more about the organization, eg. after interviewing employees of Peter Hansen Ltd.*



After the role play, the Game Masters decide, that creating the project plan took 1 consultant week - that is, one half-a-day meeting in the project group, the project managers writing of a proposal, and a shorter meeting in the project group discussing the proposal and making smaller adjustments. The meeting with the "Georg Sørensen" counts for 1 day, that is, the actual meeting and the briefing of the rest of the Project Group.

The participants are now asked to initiate the next activity. The project manager wants to make a phone call to Hans Olsen, to arrange a meeting at Peter Hansen with some of the employees, who may get involved in the project.

To see how this activity will go, the dice are thrown, and before that, the ruler has to be adjusted. The result is 07, which is in the "failure"-interval.

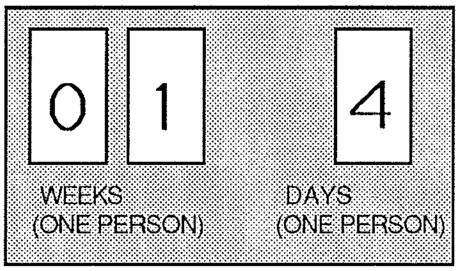


Role playing II:

The project manager phones Hans Olsen, played by one of the Game Masters. The project manager does not get through to Hans Olsen, because he is at a meeting. Hans Olsen's secretary is, however, capable of arranging a meeting on Hans Olsen's behalf. The meeting is settled to be in a week from now, although the project manager wants it to be earlier. Still, the project manager wishes to speak to Hans Olsen to discuss the arrangement of the meeting, so he leaves a message to Hans Olsen. Three days later, Hans Olsen has not phoned yet, and the project manager is about to try to again, when Hans Olsen's secretary phones him instead. She says, that Hans Olsen is sorry, but he has to cancel the meeting. Colleagues of his have just told him about a salesmen congress in Germany, that he wants to attend. Hans Olsen did not have time to phone the project manager before he left for the congress. The congress runs for three days, and when Hans Olsen gets back he will call. Six days later Hans Olsen phones the project manager, and they discuss the meeting. The project manager is annoyed, but he is trying to control it. The conversation turns out to be rather formal, and the project manager does not get much information on the organization. He does get a meeting arranged, though, in the end of the week. Besides Hans Olsen, a representative from the regional sales office, a store keeper, and the edb-manager will participate.

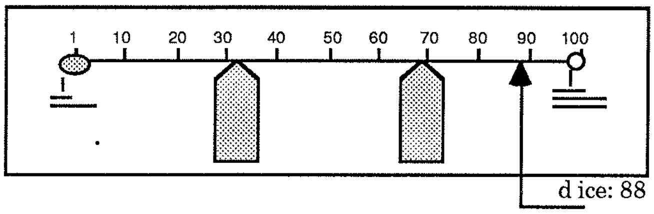
All this phoning counts for a half day on the time-table. Because the number was that low, the Performance of the project group is reduced to 13.

The Project Group is now discussing a plan for the meeting at Peter Hansen Ltd. The Game Masters start the stop watch. The Project Group decides, that only three of them are going to the meeting, those are the project manager and the two analysts. They decide that the purpose of the meeting is to get a first impression of the organization and to get some written information on the organizational structure and the division of labor. Furthermore, they are going to introduce the overall project plan, and get an agreement on the first part, the interviews.



The discussion in the Project Group lasted for 5 minutes, which corresponds to half a days meeting for 5 people, that is half a week.

This meeting is also played as a role play. But first the probability ruler is adjusted and the dice are thrown. The result is 88 which is in the "success"-interval.

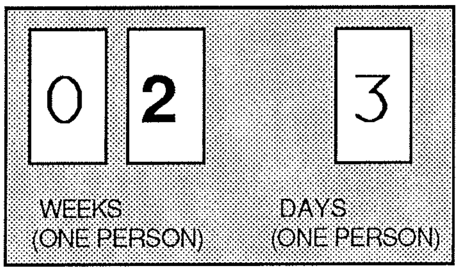


Role play III:

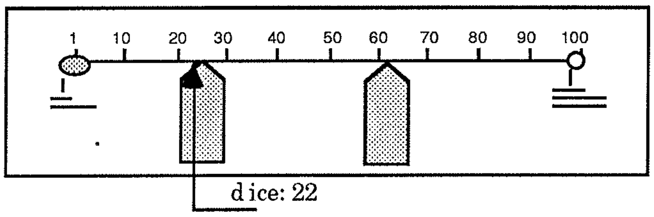
The Game Masters play the roles of all the four participants from Peter Hansen. At the meeting there is a positive atmosphere. First, everybody introduce themselves. Besides Hans Olsen, there are, Per Jensen, manager of the regional sales office in Sjælland, Jens Peter Olsen, store keeper from Næstved, and Peter Bergstrøm, edp-manager. The participants from Peter Hansen explain how the stock control system works today, and what problems there are (in the play we actually tells these things, but that would be too detailed in this presentation). The Project Group presents the plan for the project, and the project manager suggests, that a steering committee is appointed. Hans Olsen agrees upon this, and he will See to the practical arrangement of it. Then the Project Group asks for some written information about Peter Hansen Ltd.They get an organization chart and a map showing where the different plants, regional sales offices and district sales offices are placed. They use this to discuss, who it is appropriate to interview. They ask also for some documentation on the present stock control system, but the edb-department has only a users manual. The edp-manager will see, what he *can get from the firm, that developed and are maintaining the system . At the end of the meeting, the project manager asks how the decision making about the project has been, and Hans Olsen explains about decision making in the board of directors. The project manager suggests that some general information about the project is distributed in the organization, to assure that the employees know what is going on, when the interviews start. Hans Olsen would like the project group to take care of this. They accept it, but they want the information to be checked by some one at Peter Hansen Ltd. before it is distributed. Hans Olsen will do this check, unless the steering committee is already appointed. Furthermore Hans Olsen will arrange a discussion about the project at the next meeting in the collaboration committee.*

This was half-a-days meeting, with 3 people participating. They wrote a report from the meeting, so it counts for 2 days. Furthermore the meeting was a success, and the performance of the group is increased to 15.

The project group now discusses a more detailed plan for the next activities. They decide, that Ulla Knapstrup is going to write the information to the employees and check it with Hans Olsen. Furthermore, the project manager is going to plan the interviews in collaboration with John Jensen. The purpose of the interviews is to get some more general information about Peter Hansen Ltd. and especially the stock control. But at the same time they wish to use the information to make the first data flow diagrams covering the present stock control system. Finally, they decide that Ole Strandgaard and Ulla Knapstrup participate in the meeting in the collaboration committee, if the project group is asked to do so.



The meeting in the project group lasted for 4 minutes, so it counts for 2 days on the time schedule.



Ulla Knapstrup writes the information material and sends it to Hans Olsen. The dice are now going to be thrown again, to See what happens with the material, and how it is accepted at Peter Hansen Ltd.

The result of the dice are 22,which is in the "failure"-interval. This is interpreted by the Game Master as follows: Hans Olsen got the information paper from Ulla Knapstrup, but after one week he had not had the time to look it over, so he got it printed and distributed in the organization. And no more happens at this time.

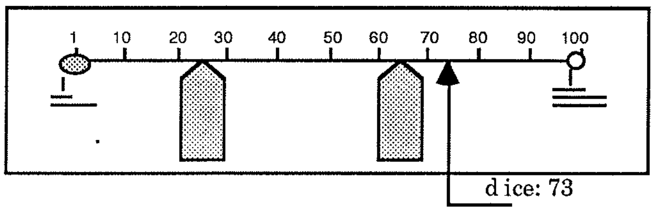
[Game Masters comment: The content of the information is OK, although the employees can not see what consequences the change may have. It leads to problems in the organization, because the employee and the shop stewards had not heard about the project earlier. These problems are running in the organization, but the Project Group does not know about them. They may show up later, e.g. in some of the interview. A bad throw for one of the interviews can now be interpretated this way: the person who is interviewed is reluctant and gives only little informations, which may be a strategy from the union. Or the the problems might lead to the calling of an extraordinary meeting in the collaboration committee. Or ...]

Writing the information material counts for 1 day. And the number 22 gets the performance down to 14.

Looking at the material from Peter Hansen, Ole Strandgaard and John Jensen decide to make 7 recorded interviews, about 1 hour each. Planing the interviews, they decide that Ole Strandgaard has the main responsibility, and John Jensen is only to interfere if he is missing something for the data flow diagram. They want to interview the following people in the following order:

- the people in one district sales office, the one in Ballerup,
- 2 - 3 people in a regional sales office, the one in Odense,
- the store keeper and maybe some other people at the stock in the plant at Næstved,
- the manager of the edp-department,
- some people from the accountancy department - to see the connection to these activities,
- some people from the strategic planning - to see their plans in this area,
- Hans Olsen, and maybe one more from the sales department.

The first interview is now going to take place. The ruler is adjusted. The dice are thrown, and the result is 73, which is in the "success"-interval.



The game masters now tell the participants, that the interview vent very well, and they summarize what information the analysts got on the present stock control system. Furthermore, the participants get some information on the problems with the information paper. Finally, they get some hints to what organizational problems, that may show up later in the project.

At this point, the first round of the game ended. In the second round, the further development of the stock-control system continues. The experience you can get from a description of a game like the present, is far from the experience you get when you participate. You miss the atmosphere, the discussions in the project group, attitudes, stresses in language, and so on.

Reflections on the game

After each round of the game there are reflections on the game, and now the audience join the discussion. The discussion deals with the Project Groups work and especially the choices they made during the game. It is in these discussions we feel that we gain from using the game. The discussions are very concrete, and the students, both the participants and the audience, join the discussions using their own judgement. Highlights from the reflections after the first round of the above game include:

Why use data flow diagrams?

The students have made data flow diagrams earlier in the course and they know that it takes time, if you want to include all details. The Project Group argues that they deliberately did not make a data flow analysis according to the book, because they would not spent that much time on the data flow analysis. An the other hand, they felt that focussing on the data flow in the organization would be good way for them to get an overview of what went on in the organization. But they did not want data flow to be the only perspective on the organization. Some of the students favour object-oriented descriptions and they start a discussion on using these descriptions instead of data flow descriptions. That discussion includes points like: What are the objects in this case? Examples of what could be revealed by making the Object-oriented description instead of the data flow description.

The Project Groups difficulties in getting the first meeting settled.

The students complain that normally you would not have that much trouble in getting a meeting settled. And that is true - *normally* you would not, but this situation was not created as a situation that is likely to happen - it was created as a worst-case, because the throw of the dice was low. And as a worst-case the situation is not so unlikely. The whole point in using the dice is exactly to give this variation in the game.

The problems that are coming up in the organization.

Some of the students form the audience complain that the Project Group did not plan any follow up on the information paper to the employees. They do not believe that it is wise only to distribute some written information. The Project Group agrees upon this, but they have found it hard to keep cool, when they realized that whenever they did anything it cost time. This leads to a discussion on professional roles. For instance: When do the Project Group know better than Hans Olsen and the others at Peter Hansen Ltd? When should they insist and when do they have to compromise?

The students from the audience suggest that the Project Group should have insisted to at least perform an information meeting, this way they the would get an impression on the different attitudes. In the current state of the game, it seems like the problems come from more fundamental conflicts in the organization. In that case, the students from the audience suggest that, Project Group regards these problems as the organizations problems and not the projects problems. This leads to a discussion on how to do that!

Thus, the reflections after the first round of the game serves the purpose of evaluating the project so far. The Project Group get a chance to discuss the problems they have felt, and the audience get a chance to give some good advice.

Discussion

Computers and Conflicts was developed for the IRIS-89 conference. Before that, we had not played the game with students learning systems development. We had, of course, played the game with some of our colleagues to try it out. At the workshop at the conference, we played the game, and after that we have played the first round of a game in teaching systems development at the institute. The next rounds are to come.

In connection to these different games, we have had discussions af some of the elements of the game and also, we have tryed different variations of the game. We do not feel that the game is fully developed yet. The problems we have discussed, have to do with breakdowns in the game. For instance if the players are confused and do not know what is expected of them, or they feel the situation is too unrealistic.

We want the game to, to some extend, seem realistic in order to support discussion like: "What could we as systems developers do if anything like this happened?". On the other hand we also want the players to get seriously involved while playing the game. The players are not supposed to reflect on the game while playing it - that comes later. The two purposes may easily conflict, for instance the dice are important when it comes to involvement, but throwing them is certainly not an act of real life systems development. This conflict has been underlying many of the discussions of the game we have had.

One thing, that has often been discussed, is the possible gab between the numbers of the Role Character sheet and the players personal qualities. For instance, a player who is actually quite good at communicating his or her ideas, may end up with a Role Character with a low value in "communicative abilities". This does not seem fair. The Project Group may choose to send this player to a meeting, that will be played as a role play, even though they would never do so if they were only looking at the Role Character Sheets. One way of solving this problem would be to leave the Basic Qualities out of the game. All the players would then have the same changes (when it comes to Basic Qualities) when the dice are thrown, and in the following role play they would use their personal qualities. Before we do a thing like that, we have to consider why we included the Basic Qualities in the first place. The Basic Qualities were intended to make it easier to discuss each others best abilities, and to take this into account when choosing who is going to do what. It is a whole lot easier to say: "We'd better send you, cause you are good at communicating", when all you are referring to are some numbers on the Role Character Sheets. One of our purposes of the game is to teach the students to use each others best abilities. Because it is obvious to do it in the game, we hope it will be more natural in real situations afterwards.

This is why we would be sorry to let go of the Basic Qualities. The problem can be minimized by strongly advising the players to choose values for Basic Qualities that matches those of their own personal qualities.

Another problem often discussed is: Who is going to throw the dice, and who is going to know the result?. It is not very realistic that before you go to a meeting, you know whether you are going to have good luck or bad luck. From this point of view it is not right to let the players throw the dice, one should rather let the Game Masters or someone in the audience throw the dice and not tell the result to the players until after the role play has ended. After having tried both variations, we have decided to stick to the first version and let the players throw their own dice. This way, the players feel more responsible for the result of the dice, and are more willing to fight hard when they have bad luck.

Another problem in relation to this is that, even though the ruler says so, the throw is not about success or failure. The Project Group can through ingenuity and hard work turn a failure into a success. This ought to be more obvious in the game. That is, it ought to be more clear that the throw of the dice is about the predisposition of the persons the Project Group is going to meet. This could be done by doing more than one throw, for instance, a throw for the predisposition of the individuals at the meeting (not the Project Group members, though), a throw for the environment, and so on.

But, we are also planning changes of the game that are driven by the wish to be more realistic. One of these is a change to make it possible to measure time in a third way, that is to keep track of the dates. The way the game is now, we can not tell whether the deadlines of the project are kept or not, since we only measure the time spent on the project - not at what date the time was spent. We have not decided yet how we are going to implement this change.

Finally, we have had a lot of discussions of the concept of Performance. Firstly, we no longer see any reason for having Performance as a group characteristic and not an individual one. This will become interesting when the same Project Group plays several different games. An individual Performance may lead to the state that one of the member of the group gets a very low Performance, which causes troubles.

Secondly, the Performance is affected by the throw of the dice. If it is a high throw the Performance increases, if it is a low throw the Performance decreases. During the game the Performance becomes an expression of how well the Project Group is doing. But what it do express is how well the Project Group is doing at throwing the dice. What the Project Group ought to be rewarded for is, when they get a low throw and as a consequence of that a hard time, and still they manage to get a good result. This way the Performance would express how well the Project Group is doing in the activities, rather than how well they are doing in throwing dice.

Finally, we do not like the one-dimensionality of the Performance. In the game, the Performance easily ends up being a measurement for wether you are winning or loosing - it becomes the points you score in the game. "Winning and loosing" in real life systems development do not have the same one-dimensionality, that is you never really know if you had a success or not. Since we do not see any way to bring a multi-dimensional concept of Performance into the game, we have decided to leave the game as it is, and then bring the problem up in the discussion afterwards.

Final Remarks

Teaching students systems development is not the only context for Computers and Conflicts. We plan to develop a similar game for teaching union representatives how to deal with technological changes. Doing this would imply only a few changes in the background material of the game.

In this paper, we have only given an overview of the game. The paper does not represent all the material it takes to play the game. As Game Masters we have a specific book about the user organization, the persons in the organization, the different conflicts in the user organization, other processes of change, and so on. In Computers and Conflicts there is no winning strategy, no matter what the Project Group chooses to do they will run into some kind of trouble. Besides using the information in the book, the Game Masters do a lot of improvisations during a game.

If you, by reading this paper, have caught interest in the game, we will gladly consider playing a game with your students.

References

When designing the game, including designing the different problems and conflicts the Project Group could run into, we used the following references:

Systemudvikling i praksis: Jydsk Telefon, MARS-rapport nr 2, june 1984

Systemudvikling i praksis: Regnecentrqalen af 1979, MARS-rapport nr 3, june 1984

Systemudvikling i praksis: Sparekassernes Datacenter, MARS-rapport nr 4, june 1984

Systemudvikling i praksis: ØK-data, MARS-rapport nr 5, june 1984

Teknologi udvalget smøger ærmerne op, HKs Komunale Landsforening, 1987

Teknologi hvordan? Det komunale Teknologiudvalg, 1987

Historier fra det virkelige liv, M.Bartholdy, B.S.Rolskov,1987

The only reference used directly in this paper is:

Dungeons and Dragons, Gary Gygax, Dave Arneson, TSR, 1972

EXPANSIVE SYSTEMS DEVELOPMENT

Ole Bisgaard
Preben Mogensen
Mads Nørby
Michael Thomsen

Department of Computer Science
University of Aarhus

Abstract

This paper presents an activity-theoretical framework for understanding systems development as an integrated part of expansive organizational development. Organizational problems are seen as being caused by contradictions, which when brought to light can lead 'users' to come up with creative and expansive solutions.

The framework forms the conceptual basis for an expansive systems development focusing on design processes and 'users' rather than on systems and developers.

The design-process is viewed as a voyage of expansive development in which the systems developers act as guides for the 'users'. In search for a systems development practice based on the framework the article discusses candidates for methods and techniques to support such a development process.

Introduction

> *Life is islands of ecstasy in an ocean of ennui, and after the age of thirty land is seldom seen. At best we wander from one much-worn sandbar to the next, soon familiar with each grain of sand we see.* [12, p.12]

With this discouraging thought, Luke Rhinehart in his 'autobiography', *The Dice Man* [12] introduces us to the crisis he faces as a 32-year old psychiatrist with a beautiful wife, two children, a comfortable life and a fairly succesful practice. His life has turned out to be boring, and he finds it increasingly difficult to accept that psychiatry is only a matter of decreasing the patients' misery and increase their productivity.

> *It had always seemed to me a quite obvious and desirable goal for therapy but, after having been 'succesfully' analyzed and after having lived in moderate happiness with moderate success with an average wife and family for seven years, I found suddenly, around my thirty-second birthday, that I wanted to kill myself. And to kill several other people too.* [12, p.13]
> ...

I was caught in a bind. On the one hand I was bored and dissatisfied with my life and myself as they had been for the past decade; on the other, no conceivable change seemed preferable. I was too old to believe that lounging on the shores of Tahiti, becoming a wealthy television personality, being buddy-buddy with Erich Fromm, Teddy Kennedy or Bob Dylan, or entertaining Sophia Loren and Raquel Welch in the same bed for a month or so would change anything. No matter how I twisted or turned there seemed to be an anchor in my chest which held me fast... [12, p.15]

The radical change in Luke Rhineharts life takes place after a wet and disastrous poker-evening. His mentor, Dr. Mann, has just left after pulling Luke's ego and psychiatric ideals apart, and Luke is walking aimlessly around the apartment wanting to do something wild. His wife has passed out in the bedroom, and all the guests have left, including Arlene, the wife of his colleague, neighbour and friend, Jake Ecstein. In the course of the evening Luke has been experiencing a growing desire to engage in extra-marital activity with Arlene, and as he stumbles over a die covered by a playing-card, he makes the decision that will change his life:

..if that die has a one face up, I thought, I'm going downstairs and rape Arlene ... a one means rape, the other numbers mean bed, the die is cast. Who am I to question the die?
I picked up the queen of spades and saw staring at me a cyclopean eye: a one.[12, p.69]

He obeys the decision of the die, and discovers that he has found a clue to a new life in leaving all power to the randomness of dice. Letting the die decide supplies him with the possibility of exploring his subconscious desires and the childhood potentials that have been curtailed and crippled by the rules and ethics of society.

In the course of time Luke turns into The Dice Man, using dice in his practice as well as in his private life and founding The Religion of the Die, which finds many followers but, naturally, also many fierce opponents. He is dismissed from the American Medical Association, ridiculed by the press, persecuted by religious groups and, as a result of his die-dictated murder of a former patient, he ends up a fugitive from the law. At the end of the book he is faced with the option of betrayal or 237 years in jail. With half an hour to decide he discusses the problem with his wife and Jake:

'Create the options. Shake the dice. All else is nonsense.'
'But I'm *worried. It's* me *that may get two hundred and thirty-seven years.'*
'Who're you?' Jake asked lazily.
There was a long pause and by now all three of us were staring into the red glow.
'Oh yeah, I keep forgetting,' I said, pulling out a green die, sitting up erect and becoming aware that I was sitting on someone's snow. 'I am [12, p.541]

The problem and its solutions

Luke Rhinehart's story contains elements of four basic ideas in the theory, we will present in this paper.

Many people will recognize the problem facing Luke at the beginning of the story from their own private life, but we will argue that the 'bind', in which a problem is acknowledged but no solutions within the given context seem to be acceptable, also typifies many problematic situations encountered in work-situations.

It may seem that Luke in his aimless wanderings accidentally stumples upon the solution to his problem. He calls it fate, but we shall argue, that situations in which solutions are found can be sought or provoked, and that the solution found to a large degree depends upon the path chosen.

The solution that presents itself to Luke is to expand his own practice by using the dice to liberate himself from the social and moral constraints of his daily life. We shall argue that situations like his call for the expansion of practice, but that viable solutions will seldomly be found through random expansion. Luke's search for a random environment throws him into a deadlock situation worse than the one he came from (although he may claim otherwise). In the end he is at odds with the majority of the surrounding society including his own professional community, and he seems to have no way out. We shall argue that if expansive solutions are to have a chance of survival in the long run, they must be based on the given practice and take the constraints of the surrounding community into consideration.

The theory

Systems development (SD) is intrinsically related to organizational development. An organizational problem is encountered. Someone decides that the solution or part of it lies in the development of a new computer system or enhancement of the existing one. The system developers are called in.

What we encounter is a problematic situation - a need for change. The problem can be more or less defined. In some situations all that is called for is programming assistance, but in others it is clear that the proposed change will have significant effect upon the organization itself and the daily work of individual members of the organization. In the latter case the task facing the systems developer is not primarily that of programming a computer system but that of guiding, assisting and/or taking part in organizational development.

Bearing the given example in mind we shall take as our starting point, that any succesful organizational development must take its outset in an understanding of the organization itself, its societal context and the daily work practice of its members. When expansive solutions are called for, inspiration must be found outside the given context, but it must be guided by an understanding of the practice which is to be developed.

On the following pages we propose a theoretical framework designed to help facilitate the understanding of practice and the development of practice. The framework is based on Yrjö Engeström's interpretation of activity theory [7], and it is presented in detail in our thesis *Systemudvikling som lærevirksomhed* [4]. In order to relate the theory to existing SD research, we first present the theoretical foundation through a paradigm-oriented discussion. Having done this we introduce the fundamental concepts of the theory, and finally we outline an SD framework based on the theory and discuss the roles of the different participants in an SD project.

An activity-theoretical framework for SD

During the last two decades SD research, in particular in Scandinavia, has been preoccupied with criticism of the socalled information-theoretical approach[1] to SD. The ever-increasing integration of computers into all areas of society has forced computer science to shift part of its focus away from mathematical theories of computation and description towards social, organizational and psychological issues. The empirical-logical paradigm, which assumes that the world and its contents can be observed, fully understood and described in logical terms, has been discarded as mechanistic and unable to account for the problems encountered in the development of systems, that directly influence the daily life. The focus has shifted away from the machine and towards the human being, and a wide array of alternative methods and approaches have appeared.

Some see this development as a beginning paradigm shift, while others take the post-modern stance and talk of parallel paradigms or paradigm cases, but none seem to have been able to pinpoint the new paradigm or give the different paradigm cases a common conceptual basis, which can encapsulate or frame the science of SD.

The theoretical foundation

Activity theory is in a way an anti-paradigm. Its roots are in dialectic materialism, and one could therefore assume that its paradigm is defined by the dialectic laws and the materialistic doctrine. There are certainly elements of both in activity theory, but in order to discuss its paradigmatic character we have to focus on its core concept – human practice.

The concept of practice in activity theory finds its roots in Marx's critizism of traditional materialism in his Thesis on Feuerbach:

[1]In his book *Systemudvikling – teori og historie i skandinavisk perspektiv* [3] J. Bansler identifies three different SD approaches in Scandinavia: the information-theoretical, the socio-technical and the collective resource approach.

The chief defect of all previous materialism (...) is that things (Gegenstand), reality, sensousness are conceived only in the form of the **object**, *or of* **contemplation,** *but not as* **sensous human activity, practice,** *not subjectively. Hence, in contradistinction to materialism, the* **active** *side was set forth abstractly by idealism - which, of course, does not know real, sensous activity as such.* Cited from [7, p.38]

The point of this quote is that objective materialism is not adequate for describing the world and human life. The world is always viewed through someones eyes and is thus interpreted. We must therefore include an understanding of the character of the interpretation – the subjective dimension which, according to Marx, is determined and shaped by human practice.

This point of view supplies us with a very useful perspective on the discussion of paradigms. A paradigm is understood as a representation of a shared practice, and as such it is subject to continued conscious or non-conscious change through the change of practice. This means that we cannot define or hope to find an absolute and fixed standpoint from which to view the world – our ideologies, models etc. will always reflect and be reflected in our practice – and consequently we do not have to accept any given truth or paradigm but can adopt and mold ideas and thoughts that seem productive to our own practice. Human practice is historically determined, and we must be aware, that our own outset in the endeavour to find and adopt new ideas is our own, historically determined practice, which shapes our perspective and sets the standard for our search. Any development of our own practice must take its outset in our own practice, and the best we can hope for is to be able to momentarily leave our practice to view it with the eyes of other practices.

This concept of practice and development has profound influence on our framework. It constitutes a liberation from the demand of development of a new paradigm or attachment to an existing, and it provides us with a practical standpoint from which to discuss issues such as communication, cooperation and mutual understanding.

As reality can only be viewed and described through the perspective of a given practice, objects in the world, such as computer systems, and models of the world, such as SD-methodologies, are determined and evaluated primarily through their use within a given practice rather than by some 'objective' qualities. The philosophy thus exerts its influence on all levels of the framework and, as will hopefully become apparent to the reader, it provides us with a consistent frame for a broader and clearer understanding of several issues in SD.

As systems developers we concern ourselves with the development of computer systems, but, as we have just pointed out, such systems are primarily to be understood in terms of their use within a given practice, and we must therefore broaden our field to include the development of practice. One could say that our unit of analysis is human practice, and our offset in the development of a framework for SD must therefore be a model of human practice.

On the following pages we present our interpretation of Engeströms model of human activity (human practice) and use this model as the basis for modelling expansive development of human activity. Having presented the cycle of expansive development we discuss its practical implications and present a few methods that seem relevant and suitable within the framework.

The models of activity and of the development of activity should not be considered fixed. In concordance with activity theory they do not claim any general validity. Their purpose in the context of SD is to inspire and expand our practice as systems developers and SD-researchers.

The concept of activity – human practice

The concept of human activity that we present here finds its origin in Leontjevs separation of human practice in three layers: operations, actions and activity. **Operations** are non-conscious reactions to external **conditions**, and **actions** are conscious and directed towards particular **goals**. While operations and actions are individual, **activity** is collective and in correspondance with a certain **motive**. As demonstrated in the following example this means that only shared practice directed toward a certain motive can be labelled activity. Activity is composed of actions that are realized through operations.

Due to the division of labour in society there is very seldom a direct correspondence between the work of an individual and the underlying motive. As an example the work of a clerk seems totally inadequate in relation to the motive of achieving food, clothing etc. Only because of the relation between the clerks work at the office and the work of others in the organization does his work have a meaningfull purpose. On the one hand there has to be a relation between the work of individuals in the organization to 'justify' a salary, and on the other hand there also has to be a relation to the work of others outside the organization in order to convert this salary into food, clothing etc. It is thus not possible to talk about the individual activity but only about the activity of the individual i.e. the activity in which the actions of the individual are contained.

The notion of activity as collective and directed towards a motive is depicted in Engeströms triadic model of activity (fig. 1).

With the use of instruments (tools, models etc.) the collective subject manipulates some object and produces an outcome which is distributed to the community according to the division of labour and exchanged among the individual members of the community according to need and governed by rules. A part of the production is consumed within the activity itself.

The triadic structure of the model reflects the relations between the different corners. Two corners stand in relation to each other only through the mediation of a third corner (e.g. the subject can only manipulate the object through the use of instruments). This means that a change in one corner will influence all the other corners, and it thus gives

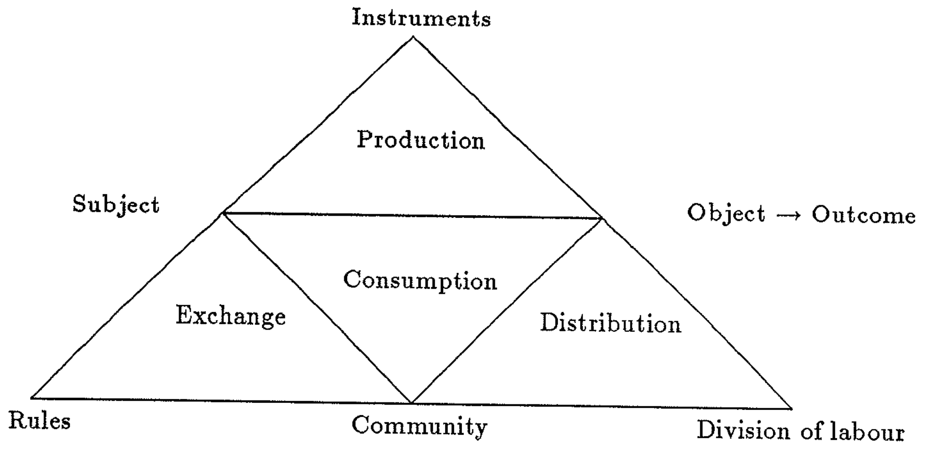


Figure 1: The structure of human activity

us a picture of the effects on activity invariably exerted by a new computer system.

An activity is held together by a motive, which is directed partly at the upholding and development of the activity itself ('organizational health') and partly at the outcome. The outcome or product is produced in order to satisfy the needs of other activities, i.e. it has to be sold in a market or be used in activities in other levels in a hierarchy of activities.

A clerks working place might be conceived as a hierarchy of activities: the clerk works in an office, which is part of the factory, which is part of ...

On the one hand, these activities are tied together by economical and cultural forces – the administrative department delivers services to the factory which in turn delivers resources, and the cultural forces emerge through participation in the activity.[2]

On the other hand, the same forces which tie together the hierarchy constitutes the basis for contradictions – contradictions between the motive of upholding and development of the activity and the motive of serving the activity of which it is part.

In the example of the administration, the contradictions could be manifested as follows. The motive of the administration[3] could be to strengthen its position in the factory as well as quality of the delivered service. The factory's motive could for example be to strengthen its position in the market. This motive could lead to a demand of a reduction of the size and increase in the efficiency of the administration.

This separation of motive, which is caused by the division of labor, represents the **primary contradiction** of the activity. Primary contradictions are permanent but not necessarily manifest as a problem for the subject. When they become manifest they can lead to

[2]Schein talks about the culture of an organization as learned responses to the problems of 'surviving', that the organization might meet [14, p.14ff].

[3]Short for the motive of the people in the organization, who perceive themselves as taking part in the activity of the administration.

a socalled **need state** in which the subject is both conscious of a need for change and vulnerable to change coming from the outside.

A change (whether it is imposed from the outside or brought about by the subject as a reaction to a need state) in one corner of the activity will cause discrepancies between this corner and other corners of the activity. These discrepancies are called **secondary contradictions.** In some cases secondary contradictions can be solved within the confines of the existing activity, but sometimes they can threaten the activity itself because it is unable to ignore the changes and unable to adapt to them. In these situations the need state evolves into a so-called **double bind**, a sort of catch-22 situation. The problems cannot be solved within the confines of the activity itself, and it will need to break its own limits through what we shall call **expansive development.**

Expansive development

Expansive development is initiated by a consciousness of a double bind situation within a part of the community. Representatives of the subject acknowledge the double bind and start looking for solutions. Within science this process is characterized by T. S. Kuhn as extraordinary science. The claim that the double bind can only be solved through expansion of the activity implies, that the clues to solutions have to be found outside the traditional boundaries of the activity itself. The clue or clues that lead to the initial expansion are called **springboards.** In principle they can appear out of nowhere as a response to the double bind situation, but, as expansion always happens from within the historically determined activity, springboards can be consciously sought out and developed on the basis of an understanding of the activity itself.

The appearance of a springboard can lead to a vision of a new, expanded activity - **the given new activity.** This does not mean, that the activity instantly expands into a new activity. The given new is first and foremost a vision held by representatives of the subject, and several steps are needed in order to implement this vision within the community. First of all the given new activity is modelled, which will normally happen through a modelling of the different components of the activity. This model then is applied within a socalled **microcosm** - an isolated group of representatives of the subject, which performs actions in accordance with the given new activity.

Modelling and application within a microcosm will naturally lead to changes in the vision, but the real challenge takes place when the given new activity is applied within the total activity system. In this phase the given new activity takes up the struggle with the old activity, a struggle which gives rise to socalled **tertiary contradictions.** The given new activity may lose this struggle, and it will invariably change as a result of the tertiary contradictions with the old activity. The result of this gradual change is called the precursor of the **created new activity**, which can be characterized by its replacement of the old activity within the total activity system.

When a new activity is introduced within an activity system, the neighbouring activities[4] will be affected. The reactions from these neighbouring activities can lead to socalled **quarternary contradictions**, which can lead to development of the other activities or an introduction of a new need state leading to a new cycle of development. The created new activity with its consolidation within the activity system thus marks the end of a development cycle that can be pictured as seen in fig. 2.

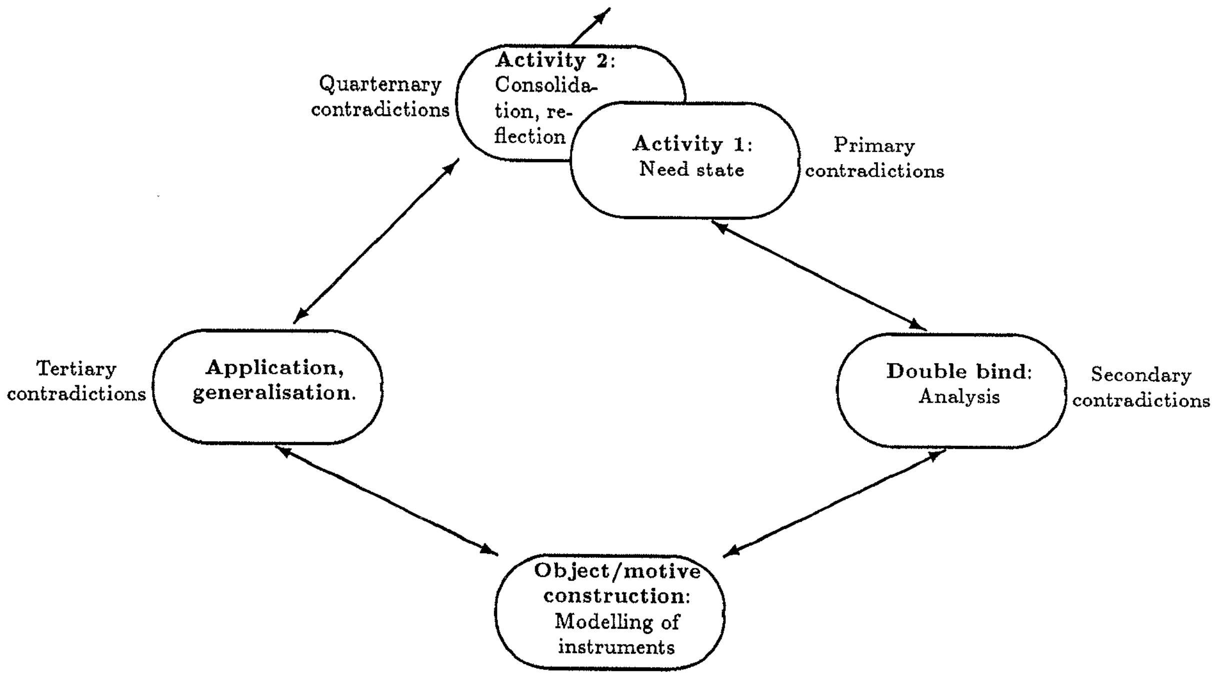

Activity 2: Consolidation, reflection
Quarternary contradictions
Activity 1: Need state
Primary contradictions
Tertiary contradictions
Application, generalisation.
Double bind: Analysis
Secondary contradictions
Object/motive construction: Modelling of instruments

Figure 2: The cycle of expansive development

The Dice Man revisited

In order to clarify the concepts presented we can apply them to the case of The Dice Man. Bear in mind though, that the example describes the need state and double bind of an individual rather than a collective subject. This means that the need state and double bind have an individual, psychological character rather than being caused by a change in a collective activity.

It is clear that Luke Rhinehart finds himself in a need state at the outset. The activities in which he takes part are not satisfying to him. According to his own description this need state arises seemingly out of nowhere. He debates the psychological cause of the depression and concludes that it is a dissatisfaction with past accomplishments and future hopes, and that it has existed and grown for a long time.

[4]The neighbouring activities are: the subject producing, the instrument producing, the rule producing and the object activity.

In terms of the activity theory we have presented, we can interpret his situation as follows: As a psychiatrist with a private practice his primary contradiction can be characterized as the conflict between his own ideal as a psychiatrist and the reality of his profession. He wants to help people, to unleash their potential, but the psychiatric reality is that the psychiatrist tries to make people accept their weaknees and to lessen their misery while living a comfortable, carefree life himself.

As long as he was in the process of education, getting married, having children and setting up his practice the contradiction was perhaps apparent to him, but not as a problem. When we meet him, he is settled but stuck in the process of writing a book on sado-masochism. He is suddenly faced with the fact that he has nothing to contribute – and the need state arises.

In the beginning he turns to Zen-buddhism and finds some comfort (happy boredom) there, but when Dr. Mann pulls his ego and ideals apart, his boredom gives way to a genuine double bind caused by the secondary contradiction between himself (a changed subject) and the rest of his activity.

His description of his situation and his unsuccesful search for solutions is a beautiful characterization of a double bind situation. There seems to be no way out until he stumbles over his own personal springboard – the die and the rape. After raping Arlene he forms his vision of a new activity and proceeds to model and implement it in all parts of his life.

The rest of the book is one long description of his attempt to apply and consolidate his vision. Being totally commited to his new way of life he combats the tertiary contradictions, although he continuously runs into moral and ethical problems. The spreading of the vision to the community however turns out to confront him with serious quarternary contradictions. At the end of the story these contradictions seem on the verge of defeating the new activity, and Luke finds himself in a new personal conflict.

In order to stress the warning against random development with random springboards and no consideration of the surrounding community we will leave our example with the epilogue from The Dice Man – a sort of twisted double bind:

> *One day when Luke was being chased by two FBI men with .45s he came to a cliff and leapt off, just catching the roof to a wild wine twenty yards below the ridge and dangling there. Looking down, he saw fifty feet below six policemen with machine-guns, mace, tear gas canisters and two armored cars. Just above him he saw two mice, one white and one black, beginning to gnaw away at the vine to which he clung. Suddenly he saw just in front of him a cluster of luscious ripe strawberries. 'Ah,' he said. 'A new option.'* [12, p.542]

Roles in expansive systems development

Having presented and illustrated the theory of expansive development we will now turn to SD in order to see how it looks in the light of the theory, and which contributions the theory can give as a framework for SD. We will do this by considering the roles of the product, the systems developer and the user in a SD project. Lastly we will discuss possibilities for supporting expansive development.

The product

What is the product of an SD process? The most common answer to that question is: 'a computer system'. Taking our framework into account we would say that in the typical case the product is **a new activity** of which a computer system is an important part.

Where and how does a computer system fit in the presented model of human activity? Well, when asked what he considered to be the object of his activity a worker at a power station at first answered, that it was the control panel. Faced with the question, 'Where is the customer?', he came up with supply of electrical power to the customers as his object and that in this respect the control panel was only an instrument. This example exactly represents our view: A computer system in use should always be regarded as an instrument of an activity, never as an object.

When elaborating on the instrument as a mediator and considering the different triadic structures in which the instrument takes part, one can identify several of the different perspectives on computer systems, which have emerged over the last ten years.

The triadic structure between subject, instrument and object (fig.3(a)), which models the actions of the subject on an object mediated by an instrument, resembles the tool perspective, where the computer application is seen as a tool or toolkit facilitating the individual's manipulation of some object.

Considering individual actions mediated by an instrument directed towards the community instead of towards an object reveals the model depicted in fig.3(b). This model resembles the media perspective[5], which regards the computer application as a means in the communication between human beings.

Focusing on the manipulation of an object mediated by an instrument performed by a group of people and not just an individual reveals the model depicted in fig.3(c). Much of the work done in the field of Computer Supported Cooperative work (CSCW) is in line with this perspective. The GROVE system[6], which support the writing of a joint paper by several researchers, can be viewed as an example of such work, when we consider the

[5]The tool perspective as well as the media perspective are described in Bødker: *Through the Interface* [5, p.128ff], where they are called tools approach and linguistic approach respectively.
[6]Described in Robinson: *Double Level Languages & Cooperative Working* [13, p.88ff].

activity with the single researcher as subject, all the researchers as community and the paper-writing as the object.

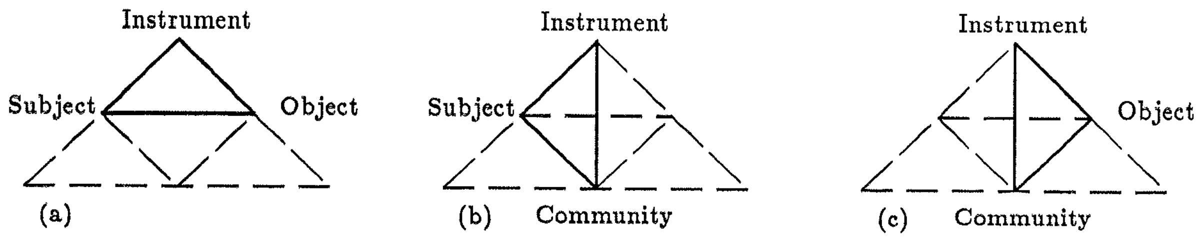


Figure 3: Different perspectives

There are other perspectives on the role of a computer system/application than the above mentioned. For example the system perspective[7], which in our terms can be seen as a perspective resembling that depicted in fig.3(a), but with the significant difference, that the goal is controlling/manipulating another activity instead of manipulating some material objects.

We will refrain from giving more examples in the hope, that we have indicated that the presented framework is strong enough to encapsulate most of the existing perspectives on computer systems/applications, and can thus contribute to a broader and more fundamental understanding of the role of computers in use.

One part of this understanding implies, that applying only one perspective is insufficient, when the task is to create a new activity. When someone uses a tool in a work task, the task and thus the tool can only be understood in a wider context (what is the meaning of just hammering nails into wood?). When someone communicates with other people, the communication must be about something. When a group of people uses a common instrument, there are individual demands.

It can though be sufficient to apply only one persective in the design of a given computer system/application, but then the triadic relations, which are not supported by the chosen perspective, must also be supported, be it with a computer application or not.

The systems developer

The systems developer is a consultant who enters a project with often little or no specific knowledge about the field of enquiry. The object of the SD-activity is the development of other people's activities with specific regard to the development of new computer systems or enhancement of existing ones. An important qualification of a systems developer is therefore the ability to master a human development process and the proficiency in describing and, to a certain extent, developing computer systems.

It is often said, that in order to be able to work properly with users, systems developers

[7]Described in Bødker: *Through the Interface* [5, p.116ff].

must spend a long time getting acquainted with the field of enquiry. The argument is that mutual understanding is a precondition for cooperation, and that mutual understanding can only be obtained through the adoption by the systems developers of the users' ways and language. There are several arguments against this position:

- It is an illusion to think, that a systems developer should be able to acquire proficiency in every field with which he comes into contact.
- If a systems developer should acquire this proficiency, he will have spend so long within the field, that he will be unable to see the forest for the trees. In terms of the activity theory he will have entered into a new activity and at least in part have absorbed its motive. He is therefore bound by the activity and biased by its motive.
- Mutual understanding is to a certain extent necessary, but only to a certain extent. The framework tells us that expansive development takes place as a result of contradictions, and it is thus driven by conflicts. The conflicts that can arise from a lack of mutual understanding can often be productive in situations where new solutions are sought. Mutual understanding is in the strictest sense equivalent to mutual activity, and an activity which is never confronted with (elements from) other activities will find no inspiration for development.

These arguments lead us to disregard the idea of mutual understanding as a prerequisite of SD and instead take up Engeström's idea of **heteroglossia**. In an SD project the systems developer is only one out of many voices that influence the symphony of development. The systems developer has his own motive and his own language, and he is confronted with the demands of the development of a computer based system, whereas the practitioners are only interested in the system insofar as it upholds and enhances their own activity.

It is a question of balance – of entering and understanding a field of enquiry whilst still standing on the outside with a fresh viewpoint and provocative tools and ideas. The systems developer acts as a guide on the practitioners voyage through the development cycle, but the actions are taken by the practitioners themselves, and in some instances the systems developer acts as the devil's advocate in that he does not reveal his knowledge but provokes the practitioners to find out for themselves.

The user

The degree of user involvement in SD projects has been intensely discussed in Scandinavia within the last decade or two – should SD be done **for**, **with**, or **by** the users?

In the previous section we have sketched how we see an SD project as a voyage through a development cycle. What you have at the beginning of such a cycle is some practitioners with a problem, which causes them to call in a systems development team to help them develop their practice. It is therefore not a question of whether to involve the users or not (in fact there are no users at this stage as there is no system), it is a question of delineating the activity which embody the problems.

As we have pointed out a given practice can only be developed from within. Inspiration can come from the outside, but the actual development has to be done by the subjects of the activity. This does not mean that the users have to develop their new instruments, but they have to discover the given new, not be instructed in it. The systems developer is a guide or a devil's advocate and thus takes active though indirect part in the development. The approach therefore mainly resembles development with the user, but without the emphasis on mutual understanding.

Towards expansive systems development

After having presented a conceptual framework for understanding expansive development of human activity and discussed the roles of people and computer systems in an expansive systems development, we now address the question: Why and when should expansive systems development take place and how can we support it?

Why and when expansive systems development?

What distinguishes expansive development from other forms of development/change is not that it takes its departure in a need state, but the character of the need state. It is essential to understand that many problems that arise in a given activity can be solved within the structure of the activity, and therefore do not require expansive solutions. As long as the problems are on the levels of operation and action they can often be solved by innovations which change the operations and actions performed within the activity but which preserve the structure of the activity (as a whole). If the participants of an activity agree upon the slowness of a word processing program as a problem, an immediate solution might be to buy a better program or faster machines (assuming no financial problems).

Only when the problems we face connect to the very structure of the activity system and thus occur on the level of activity may we talk of the need for an expansive solution. The point of departure for expansive development is a need state where the activity embodies contradictory interests – no matter which potential changes (e.g. a new computer system) one can think of, it will contribute to the resolution of conflicts for some, but at the same time reinforce conflicts for others.

This type of situation often arises when a new computer system is to be introduced into an activity – large changes in the instruments implies large changes in the use of them. The introduction of new computer systems often changes the way the daily work is performed, the division of labour between departments as well as between individuals, the rules to follow, the working groups etc. The whole activity is to be changed, and the contradictory interests of different groups become manifest.

When these contradictions become apparent the situation is characterized as a double bind, which can not be resolved within the activity. The resolution of a double bind therefore demands a change in the structure of the activity – there is a need for transcendence, breaking the rules or, as we call it, expansion.

How to support expansive systems development

An SD project carried out in accordance with our understanding of expansive development has an earlier point of intervention and a later point of withdrawal than a traditional SD project. In traditional SD the systems developer is confronted with a more or less defined problem, describes and constructs the system according to some demands and delivers it for testing and approval. With the advent of user involvement and prototyping systems developers have been able to deal with more loosely defined problems, but so far both user involvement and prototyping have had an ad hoc character due to the lack of general theories encapsulating these aspects. User-involved follow-up on the use of delivered systems rarely exceeds maintenance and error correction.

We will now try to sketch some means of supporting expansive systems development.

Defining and working with the problems

As mentioned above an expansive systems development takes its point of departure in a need state. It is therefore important to analyse the problems and needs that have given rise to the development project.

The purpose of the analysis is twofold: On the one hand it helps the systems developers to define and understand the problem to be solved, and on the other hand it should lead the practitioners to a clearer understanding of the problem and a consciousness of the double bind. In order to encourage expansive development the systems developers have to identify and bring to light the existing double bind or provoke it on the basis of the need state, the contradictions and the proposed change.

Since part of the purpose of the analysis is to provoke a double bind, it is important that the practitioners take part in the analysis or at least in a reconstruction of it. In some cases the analysis will lead the systems developers to the formulation of a hypothetical model for a new activity, but it is essential that this model is not imposed upon the practitioners. A precondition for expansive development is that it takes place from within the activity and is not imposed from the outside. The hypothetical model should act as a help to the systems developers in their guidance of the practitioners through the next stage of the developmental cycle.

In order to bring the contradictions in the activity to light the analysis has to focus on the theoretical and practical models employed in the activity and on the ability of the activity to react to changes. In this context a detailed analysis of the historical development of the activity and of its models is necessary. An insight into how the activity has previously dealt with secondary contradictions can help the systems developers to choose a course of action and avoid obvious strategical errors, and a study of models can often reveal inconsistencies or contradictions between formal or verbalized models and the models that are actually being used i.e. between theory of action and theory-in-use[8]. These inconsistencies can give a clue as to the nature of the double bind.

Verbalized models and their development can be studied in written documents, such as rules and strategies, and in existing computer systems. Theories-in-use are more fundamental than verbalized models in the sense that they reflect the basic assumptions of the practitioners. The cause of a double bind situation can often be found in the practitioners' lack of understanding of their own basic assumptions, and a revelation of the theories-in-use can therefore often demonstrate the secondary contradictions to the practitioners. Typically such theories will be reflected in the metaphors and actions of the practitioners, and the systems developer can use an analysis of these theories to demonstrate their flaws in order to make the practitioners aware of them and of the need for change.

There are many ways to study and question theories-in-use. Bearing in mind that the main aim at this stage is to reveal or provoke a double bind we will focus on methods that provoke breakdowns[9] in the theories-in-use. A few examples of what we might call active use of lack of understanding are given below.

- Let the practitioners tell stories about very concrete experiences from their activity. It is important that the practitioners try to describe their actions not the intentions behind them. This approach can possibly reveal some of the (tacit) 'theory-in-use'.[10]
- Create alternative views of the activities. This can for example be done through the use of metaphors to create breakdowns in the practitioners understanding of the present activity.[11]
- Confront the practitioners with views from other parts of the activity system in order to shed some light on the (primary) contradictions and cause breakdowns in their understanding of the activity.

Another way of provoking double binds is to let the contradictions come to light through actions. If the need state has arisen as a result of the threat of new technology, one way of clarifying the need for an expansive development of the activity as a whole could be

[8]The terms are introduced by Argyris and Schön in: *Organizational Learning: A Theory of Action Perspective* [2].

[9]Our use of the concept 'breakdown' is based on *Understanding Computers and Cognition* by Winograd and Flores [16].

[10]A discussion of this point can be found in Argyris and Schön: *Theory in practice* [1, p.39ff].

[11]This is discussed more thoroughly in Kim Halskov Madsen: *Breakthrough by Breakdown* [10], *Sprogbrug og design* [11], and Donald A. Schön: *Generative Metaphor* [15].

to simulate the realization of the threat. By implementing a simulation of the proposed system and letting the practitioners perform actions on the new system but in accordance with the old activity the systems developers can demonstrate the flaws of both the old activity and the proposed solution. We have labeled this sort of experiments **'provotypes'** – experiments with the purpose to provoke and thereby make contradictions manifest.

We know of no examples of such experiments, but visiting factories, that have experienced changes similar to the proposed, and making scenarios (e.g. mock-up simulation) [12], where actions according to the potential new activity can be performed, resembles 'provotypes'.

As mentioned, the result of the analysis should be, that the selected practitioners are conscious of their double bind and perhaps have a vision of the given new activity. A double bind is a precarius situation, as an unresolved double bind can lead to regression or directly destructive behaviour, and it is therefore now the responsibility of the systems developers to assist the practitioners in the expansive resolution of the double bind through the finding of a springboard and the elaboration of a model for the given new activity.

In search of a springboard

The generation of springboards follows as a logical extension of the provocation of double binds. As a springboard is facilitative in the expansive development of an activity it must come from somewhere outside the activity, but at the same time it has to relate in a productive way to the existing activity. Thus the analysis of the activity is essential to the search for a valid and productive springboard. The springboard will play an important role in the formation of the model of the given new activity, and the choice of direction of search for a springboard is therefore a crucial point in expansive development. If the systems developers have formed a hypothetical model of the new activity, this model can guide them in their choice of direction, but they also have the possibility of trying out different directions and methods in order to give the unexpected a chance. The latter strategy is actually the most acceptable, since there is no reason to assume, that the systems developers have come up with the optimal solution. The systems developers should see themselves as facilitators of images and creativity rather than as creators.

The springboard itself is not a vision of a given new activity. Its primary function is to shed new light upon the old activity in order to facilitate the generation of a vision. The vision will thus in some way be a synthesis of the springboard and the old activity.

Where the aim of metaphors and provotypes as breakdown-creators was provocation and highlighting of contradictions, the focus now is turned to constructive metaphors and prototypes. The flaws and inconsistencies found and revealed during the analysis can now be brought to constructive use, as the practitioners will be motivated and aware of the importance of metaphors, games and instruments. Below we will sketch some possible techniques:

[12]Both examples were used in the UTOPIA-project, see e.g. Bødker et al: *A Utopian Experience* [6].

- Method of Little Men, which Engeström describes as a method originally introduced by Altschuller. The idea of the method is, that the inventor takes the place of the object – looking for a solution from the position and viewpoint of the object.
- Future Workshops.[13]
- Simulation games, where the players simulate the different components of the activity can be used as tools for experiments with alternative activity structures.
- Exploratory prototyping[14] where the focus is on exploring the fruitfulness of different designs of the instrument of the activity.

The purpose of the use of all these techniques is to create a vision of the given new activity. The vision of a new activity is one thing, another is how it is actually implemented, when the practitioners start performing actions in accordance with the given new activity.

Manifestation of the given new

Activity is practice, and though we can possibly model the different components of an activity, the activity itself can only be modelled through application in practice. This modelling is accomplished through the formation of a microcosm, in which selected practitioners commence to perform actions in accordance with the given new activity – or rather in accordance with the models of the different components of the new activity. The idea is that the microcosm should propagate outwards into the real activity system and thus dissolve itself gradually, and it should therefore from the outset be composed of a mixture of practitioners, who contributed to the formulation of the given new activity, and other representatives of the same subject. The practitioners, who created the vision of the given new activity, must be assumed to be willing to fight for it. With the inclusion of other representatives of the subject the vision will be exposed to its first challenge within the activity system. It should be noted though, that the microcosm at least at the outset consists of only representatives of the activity itself and not of the surrounding activity system, and that the actions performed are initially only simulations of real world actions. This strategy is chosen in order to avoid the appearance of premature quarternary contradictions.

When the given new has had the chance to gain some substance it is transplanted to a real organizational setting. This does not mean that the microcosm should just develop into a normal pilot site, understood as a part of the old organizational setting with the task of testing the technical functioning of the computer system. The microcosm is supposed to become a real organizational testbench for the new activity with the task of testing the

[13]Future workshops are described by Robert Jungk and Norbert Müllert in: *Håndbog i fremtidsværksteder* [9].

[14]Different approaches to prototyping (e.g. exploratory prototyping) are described in Christiane Floyd: *A Systematic Look at Prototyping* [8].

new division of labour, the new communication, the new rules, the new instrument etc, which often means that the struggle with the old organizational hierarchy is intensified.

As experiments with the given new activity take place within the microcosm the models are changed and the microcosm is gradually expanded outwards, first through a transplantation into real world surroundings and later through the expansion to all practitioners involved. As described above this is where the given new activity will have to fight against the old and against the surrounding activities, and as a result of this fight the created new activity is developed. As the practitioners will be standing 'within' their own vision and therefore may be unable to see the consequences of the new activity and of the struggle taking place, and as the created new will invariably contain unexpected elements which can demand a remodeling of different components of the new activity (notably the instrument), it is essential that the systems developers remain involved or at least in contact in these stages.

Conclusion

We have presented a framework for SD and tried to justify, that it encapsulates and provides a frame for understanding a large part of the field. The framework supplies us with a general theory of development which we have found applicable within computer science, and it aids us in understanding the roles of the involved parties in a development project. Furthermore it serves an important purpose in placing the different methods and perspectives from our field in a broader context. Through the eyes of the theory the methods and perspectives appear as complementary rather than contradictory, and the framework thus helps us to understand, when and where to use which methods and perspectives.

Addressing the subject of the conference we would like to end with a definition of creativity expressed in the form of a question. One of the two basic problems forming the outset of Engeströms enquiry is the elusiveness of expansion, which can be metatheoretically expressed as follows:

> *How can a structure generate another structure more complex than itself?*
> [7, p.29]

References

[1] Chris Argyris and Donald A. Schön. *Theory in Practice: Increasing professional effectiveness.* Jossey – Bass Publishers, San Francisco, California, 1974.

[2] Chris Argyris and Donald A. Schön. *Organizational Learning: A Theory of Action Perspective.* Addison-Wesley Publishing Company, Reading, Massachusetts, 1978.

[3] Jørgen Bansler. *Systemudvikling – teori og historie i skandinavisk perspektiv.* Studentlitteratur, Lund, Sweden, 1987.

[4] Ole Bisgaard, Preben Mogensen, Mads Nørby, and Michael Thomsen. *Systemudvikling som lærevirksomhed – Konflikter som basis for organisationel udvikling.* DAIMI IR-88, Aarhus University, 1989.

[5] Susanne Bødker. *Through the Interface – a Human Activity Approach to User Interface Design.* DAIMI PB-224, Aarhus University, 1987.

[6] Susanne Bødker, Pelle Ehn, John Kammersgaard, Morten Kyng, and Yngve Sundblad. A utopian experience: On design of powerful computer-based tools for skilled graphic workers. In Gro Bjerknes, Pelle Ehn, and Morten Kyng, editors, *Computers and Democracy.* Avebury, 1987.

[7] Yrjö Engeström. *Learning by Expanding.* Orienta-konsultit OY, Finland, 1987.

[8] Christiane Floyd. A systematic look at prototyping. In R. Budde, K. Kuhlenkamp, L. Mathiassen, and Züllighoven H., editors, *Approaches to Prototyping.* Springer-Verlag, 1984.

[9] Robert Jungk and Norbert Müllert. *Håndbog i fremtidsværksteder.* Politisk Revy, Copenhagen, Denmark, 1984.

[10] Kim Halskov Madsen. *Breakthrough by Breakdown: Metaphors and Structured Domains.* DAIMI PB-243, Aarhus University, 1986.

[11] Kim Halskov Madsen. *Sprogbrug og Design – sammenfattende redegørelse.* DAIMI PB-245, Aarhus University, 1988.

[12] Luke Rhinehart. *The Dice Man.* Grafton Books, London, 1972.

[13] Mike Robinson. Double level languages & cooperative working. In Gerard de Zeeuw and Ranulph Glanville, editors, *Support, Society and Culture.* Department for Andragology, Amsterdam, 1989.

[14] Edgar H. Schein. *Organisationskultur og ledelse – et dynamisk perspektiv.* Forlaget Valmuen, Copenhagen, Denmark, 1986.

[15] Donald A. Schön. Generative metaphor: A perspective on problem-setting in social policy. In Andrew Ortony, editor, *Metaphor and Thought.* Avebury, 1979.

[16] Terry Winograd and Fernando Flores. *Understanding Computers and Cognition – A New Foundation for Design.* Ablex Publishing Corporation, Norwood, New Jersey, 1986.

Creativity according to the Scandinavian tradition of system development research. A contribution to a debate

Gro Bjerknes & Tone Bratteteig
University of Oslo, Dept. of Informatics
POBox 1080, Blindern
N-0316 Oslo 3, Norway

The working conference about "Creativity in System Development" in Skagen has been an opportunity to discuss creativity in system development and system development research. In our opinion creativity is a new buzzword in the Scandinavian research community, and we want to throw light on this phenomenon.

The paper discusses the notion of creativity, and concludes that talking about creativity does not contribute to any new knowledge about system development to anyone. Nevertheless, the concept may be used in arguing for professionalization of system development work and in marketing the Scandinavian reseach tradition to other computer science communities. In this case we need a definition of creativity that fits to the Scandinavian tradition.

What does creativity mean in system development research?

The concept of creativity has a positive value, and is connected to novelty, to progress and development. In a common sense interpretation of the word, creativity is connected to adding new ways of functioning to existing things. Everyday creativity means using the available resources in new ways, and by this utilize the resources in a way that makes it possible to carry out the tasks to be done. In this way it may be creative to use a shoe for a hammer. Defining creativity as finding many solutions to a problem is most clearly expressed in psychology. However, creativity as forming and being rich of ideas is not always connected to solving problems. Seeing new connections and making new expressions is more connected to understanding problems than to solving them. New insight into a topic may give new possibilities for action.

Lately, creativity has been introduced as a topic in research milieus. The notion is used to discuss system development which is carried out as a part of research projects. The discussion is more related to the system development activities than to the research methods. However, in everyday system development the possibilities for being creative - and talking about creativity - is restricted[1]. On the other hand side, we think many of the activities in system development contain creative aspects, even though only some of them are recognised as creative in the general discussion about system development and creativity.

In the following discussion we distinguish between creativity concerned with the system development process and the computer system respectively. Creativity as an aspect of the process concerns how we apply new techniques for cooperating with users in the system development process. Creativity connected to the computer system concerns how new computer technology is used to develop products.

Creativity recognised in work styles in system development.

It is easy to recognise arranging scenarios and future work shops as creative work styles in system development[2]. These are techniques for gathering a group of users in order to involve them in discussions about what they would like their future to be. Scenarios and future work shops are called "games" in order to distinguish the discussions from trivialities of the daily work[3]. The concepts of game and playing emphasizes that the activities in the work shops are simulations of real world activities. The game could be supported by some technical equipment, as means to strengthen the fantasies and visions of a new future and making them vivid by simulating future life and use of new technology. The equipment should be fancy compared to existing technical equipment in the work place, but prototypes and mock-ups may be sufficient to stimulating fantasy.

In order to play you have to create a setting different from real life, what we may call a lab-setting. If you want to play a game related to work, you have to pick the parts of work that may be isolated from the work setting. However, if cooperation and communication in real everyday work is the main topic to be addressed, it is impossible to extract and

[1]Cf. Munk-Madsen&Thaysen (1989)

[2]As far as we know Kensing (1987) was the first to introduce this kind of techniques in the Scandinavian research community. However, he does not talk about creativity in this connection. Lately these kinds of techniques have explicitly been connected to creativity, cf. e.g. Ehn (1989)

[3]The notion of game probably stems from Wittgenstein (1967). In Wittgenstein's philosophy all rule-based activities are games. A game is a situation in which it is important to follow the rules. This in turn makes it possible to isolate the game from the rest of the world.

isolate parts of work that are relevant for the daily cooperation trouble. In this case the lab-experiments will not be creative in relation to solving the cooperation problem, even though this may be simulated through role playing.

Playing is associated with creativity when grown up people are doing it. In contrast to children's playing, we always add rational explanations for adults playing. Playing with the computer system is seen as a way of exploring the technology, and the intention of playing is to improve the technology. In this way creativity legitimates using the work hours to play with a machine.

The notion of play make us associate fun, and of course work may be fun. We think, however, that we should be careful to use the concept "play" when we talk about doing system development. System development is directed towards a serious change in life for many people, changing the organisation and the work conditions. We should be careful to speak about analysing and changing somebody's work as games[4].

Work styles in system development that normally are not considered to be creative.

It is usually not considered to be very creative to use well known tools and techniques, based on established and common knowledge in system development. Nevertheless, we think it may be creative to use old fashioned means for analysis, like system description techniques. Both scenarios and system description techniques are ways of using existing understanding of the organisation as a basis for suggesting alternative future solutions. Thus, well-known system development techniques also may support our creation of a future.

The techniques recognised as creative focus on learning by doing, experience and tacit knowledge. Touching objects and looking at pictures gives emotional experiences to users and computer scientists. Hands-on experiences contribute to a deeper comprehension of a specific work situation and the skills required in work. Looking at pictures make us keep our feeling and spontaneity because most of us do not know how to "read" pictures.

Literature and descriptions do not give the same kind of personal feeling to the spectators: Concepts and text calls for reflection. Describing by means of formal description techniques always implies a certain amount of abstraction and verbalisation. This can help giving an overview of an organisation, including expressing different perspectives and eliciting conflicts. Thus old fashioned system descriptions encourage discussions with the

[4]Cf. Bjerknes&Bratteteig (1988)

users about general matters in the organisation[5]. If you want to emphasize this, you may even deliberately exceed the limits of the description tools, and use them in new ways, different from how they are supposed to be used. Using system description techniques for learning and communication makes visible the creative aspects of the description process. However, we want to stress that making a correct system descritpion involves creative thinking by the fact that any description of an organisation is input to the process of changing the organisation.

Creativity connected to computer technology.

It is accepted that it is creative to extend the limits of existing technology. One example of a technical innovation is combining different kinds of technology in computer based multi-media applications.

An application that involves some new technology, or combination of quite new technologies, is considered to be creative. Thus, simulating black boards in computer technology by making computer application for computer supported cooperative work, is seen as creative even if this may be seen as an implementation of old technology. The point is that it is not old computer technology: The creativity lies in transferring the black board facilities from one technological medium to another[6]. We should note that creativity here is connected to the computer technology itself, and that some kinds of computer based innovations are themselves considered to be creative.

In addition, exploring new kinds of application areas of the technology is recognised as creative, as e.g. new computer-based tools for system development. A new application area may in turn lead to development of new kinds of computer technology, to which the concept creativity is connected.

Use of computer technology that is not considered to be creative.

We think creativity is a siutable label when you know the craft of programming so that you may extend the rules and limits apparently set by the technology. A good programmer presses some of the limits of the technology as much as s/he can at the same time as s/he keeps other limits in order to secure that the total system will work. Let's say that your application generator is line-oriented and only allow you to present the datafields from a record in a sequence. You may accept this limit - or you may use your creativity to give the users an interface that presents the data in columns. As object-orientation and

[5]Cf. Bjerknes et al (1987) and Bjerknes&Bratteteig (1987b)
[6]Cf. for instance the many examples of simulation of black boards in CSCW (1988)

windows already exists, presenting the data in a record by columns is nothing new. Thus a look at the user interface does not tell of the creativity (or the effort!) of getting a line-oriented terminal behave like it is object-oriented. The creativity doesn't show in the product.

In this way it may be creative to make new applications without using new kinds of technology. In the Florence project, we exceeded the limits of application and report generators in order to build a computer system that was better suited to the use context. We used "ordinary" computer equipment when building a system supporting nurses' cooperative work[7]. However, as the computer system itself is not an expression of any new ideas within computer science, it is not considered to be creative.

We think the same kind of creativity may be involved when reusing old programs, maintaining them, and adjusting them to new applications. This kind of work normally is called maintenance, and many system developers regard maintenance as boring routine work. When using "old-fashioned" computer technology the creativity involved is not connected to novelty in technology itself. Rather, creativity is connected to how an organisation may utilize the possibilities of electronical data processing in the chosen application area.

A short discussion about creativity.

We have argued that some parts of system development are considered to be creative, while others are not. Of course nobody states that it is creative to use Hyper card while it is boring routine work to maintain old Cobol programs. The point is that when creativity is discussed, only some parts of system development work are mentioned. The remaining parts are implicitly defined as not creative, although nobody explicitely says so[8].

A focus on creativity tends to isolate some activities or phenomena, and to discuss them detached from the system development process. This tends to give creativity a value of its own. The focus is displaced from the goal of the system development process to the activity (or phenomenon) considered to be creative. If being creative is important, you may want to organize a future work shop even if the users would have benefited more from discussing organisational matters - which are excellently supported by ordinary system description techniques. From the same reason you may want to build a multi-media application, even if the users' needs are more directed towards dispersal than collection and presentiation of information.

[7]Cf. Bjerknes&Bratteteig (1987a)

[8]What is not talked about is just as important as what is on the agenda, cf. e.g. Bachrach&Baratz (1962)

We don't see any reason for emphasizing creativity because so far the discussion has not brought new insight to system development. Anyhow, it seems that the notion of creativity may be used to market some new techniques and technologies. On this background we will discuss which notion of creativity could be used to market the Scandinavian research tradition - and the raison d'être of system development as a discipline.

Creativity as a means for marketing the Scandinavian research tradition in system development.

We think all the above listed activities in the previous section contain creative aspects. Nevertheless, we feel that the existing notion of creativity is too much oriented towards novelty in work styles and computer systems. We think this notion of creativity is too narrow, and our arguments for this view are connected to our tradition of research in system development.

In the Scandinavian tradition of system development research, the focus has been on the users and the use context. One way of enriching the system development tradition was - and is - to bring in ideas from the humanities and the social sciences.

When the Scandinavian tradition was emerging, it was important for the members of the tradition to show that they were different from the rest of the community of computer scientists. They formed a partnership with the social scientists in order to bring more knowledge about organisations and people into the research on system development. The members of the Scandinavian system development tradition then experienced conflicts between the engineering part and the social responsible part of their interests. For some years it was not accepted to be interested in technology itself: the critical sosial scientist view on computers was adopted. As the tradition have matured, the social scientist views have become a part of the background for a widened concept of system development.

The small group of social responsible computer scientists has grown by the years. It is not that important to defend a settled tradition against external enemies, thus the members may start a dialogue with members of other computer science communities. The research tradition emphasizes the integration of the technical and the organisational sides of system development. It's time to practise the engineering part of ourselves again. We think that some use creativity as an alibi for being interested in technology.

In the beginning the Scandinavian research tradition focussed on trade unions and how to make strategies for encountering the new technology. However, strategies are of little use if the technology installed had some features the trade union did not accept. Therefore, the trade unions also should know of alternatives to the existing computer technology. Researchers from the Scandinavian research tradition tried to build computer systems from the trade unions' point of view[9]. In order to do this, it was necessary to try out ideas in a smaller scale, at specific work places. Instead of trade union strategies specialized computer solutions were developed. However, a computer system for one occupational group may have serious consequences for other occupational groups. The background of collective resource turns out to be problematic when it comes to system development practise. Nevertheless the researchers design alternative applications for different occupational groups, and by this utilize their technological creativity in the research.

The Scandinavian tradition has been a critical tradition occupied with discovering the basic values of current system development methods and techniques. It should also be said that most of the methods and techniques have been rejected due to their close relation to the systems perspective. Anyway, even the most critical system developers admit that having some methods and techniques are better than having none. Thus there has rised a need to make the critique of methods concrete by offering alternatives based on a different set of values. The Scandinavian tradition has always emphasized user participation, and considerable effort has been used in developing methods and techniques for building computer systems that fit people's work. For many years we have used techniques from psychology, anthropology, educational science etc. in order to extend the ways of cooperating with users. Recently these techniques have been labelled creative: it seems to be more creative to use well established techniques from other disciplines than from our own field.

In the Scandinavian tradition the focus has been on the relation between the work situation and the computer system. Therefore the researchers from this tradition are interested in research projects that concern applying computer systems in work, including computer systems for new application areas.

Product oriented researchers in computer science aim at improving products for existing markets, and finding new markets by discovering new application areas for computer systems. This is particularly important for the research going on in larger computer manufactorers that develop new products to sell. In Norway, the research councils tend to

[9]Cf. Bansler (1987) and Ehn&Kyng (1987)

support development of new products by supporting collaborative research projects between computer manufacturers and research institutions. This means that research milieus try to form partnerships with computer manufacturers in order to get research fundings. As the Scandinavian tradition is not particularly product-oriented there are some difficulties in getting fundings - unless you manage to relate the reseach to a product.

Both cooperation and creativity may be used to name special characteristics within Scandinavian system development research. Both of the words seems to indicate a combination of the Scandinavian thinking with new kinds of technology, which will make the idea exchange between the Scandinavian researchers and other researchers easier. We think this is an important opportunity for applying Scandinavian research results in a wider area. In addition, we think that the focus on creativity is a strategy for Scandinavian reseach in system development to keep on being something special: We claim to have a tradition of working with the users that is somewhat more creative than other traditions. Creativity is thus a means to maintain the distinction between the Scandinavian research tradition and other research traditions in system development. It is important to be something special in contrast to being in the periphery of the western Europe and US. We think that the researchers belonging to the Scandinavian system development tradition are proud of belonging to this tradition, and that they (we) want to continue to work on the basis of the established values.

We see, however, a danger of forgetting the basis of our research tradition when emphazising creativity. In giving that much attention to new technical products, we reject the Scandinavian system development research tradition that focuses on process, not product. By paying too much attention to single, isolated techniques, we may loose sight of the total complexity of the system development process. The generalizable results from almost every Scandinavian research project has to do with the process seen as a whole, not with products or a few isolated situations. Forgetting this is a too large price to pay for playing with the big guys.

Creativity as a means for marketing system development as a discipline.

In the middle of the 70ies computer science competence was a scarce resource. This was e.g. reflected in high wages and low requirements of education for getting a position. As time has gone by quite a number of unsuccessful computer systems has shown up, and quite many projects have exceeded both time and cost limits. This has lead to attempts to increase productivity and to control system development projects.

In general, one means to increase productivity and to control an activity is to standardize it. This has also been done in system development.

Application- and report generators are means to standardize the programming activities. A number of "usual" administrative applications may be more effective programmed and maintained by the use of generators. However, they do not reduce the time spent on analysis, and in many cases the use of generators is not recommended[10].

Quality assurance techniques are mostly used for controlling that requirements to the computer system are met and to enhance coordination and division of labour. It adds to standardization by introducing routines for project management, e.g. by requiring documentation for controlling that (intermediate) goals are reached.

By reducing system development to routine tasks, it is possible to employ less educated people to do system development, thus expences of wages could be reduced. This has made some people claim that system developers as such seem to become superfluous. One way to meet this (possible) challenge is to emphasize the parts of system development work which are difficult to formalize and automate. Arguments to support this position are that organisations are different and thus need tailored computer solutions, and that intuition and creativity are important aspects of system development work.

Creativity as part of the non-formalizable activities in system development differs from talking about creativity in research. In research, creativity is connected to the novelty in using techniques or technology. A broader understanding of creativity may in turn be used by researchers to legitimate the research, in two ways.

The first line of reasoning is that the Scandinavian system development research tradition takes basis in practical system development work. An important research goal is that the research should be useful outside the scientific community. As earlier said we think that there are many creative aspects of system development and system development research, connected to the use of existing system development methods and techniques and to existing computer technology. System development can not be totally industrialized, and the arguments for this position are that every organisation and every system development project is unique and non-routine (at least to some extent). Creativity is an important

[10]According to Budde et al (1989) application generators are not recommended in cases when the users themselves change the database, when the database is distributed or in case of extensive applications.

aspect of system development work, as a consequence of focusing on differences. Research in system development thus becomes important in order to gain more knowledge and to improve system development activities.

The second line of reasoning is that research should be a basis for education which should make the students of system development fit for working life. Stressing the importance of creativity in everyday system development should have some implications for system development education. Creativity is a means for controlling the education, as an argument for the contents and form of the education. Certifying and controlling the education may in the long run lead to professionalization of the occupation[11]. Professionalization of system development means that there will be no need to legitimate the activity itself. As long as the society approve system development, it should be quite easy to argue for research in and about the field.

Conclusions.

We do agree that it is creative work to make new methods and new computer-based tools. Creativity defined in this "narrow" sense may be used to point at the Scandinavian research tradition as something different from other traditions by our experiences with using techniques from other branches in work. The integration of social science techniques in system development makes our tradition able to bring something new to other system development research communities.

We react, however, against that the notion of creativity is that closely connected to new techniques and new tools. If we want to use creativity in order to protect ourselves, we should broaden the scope of the notion. We should define creativity in line of the Scandinavian tradition in the sense that we take basis in the use of techniques and technology, instead of looking at creativity as an isolated matter. Creativity means using available resources in new ways in order to carry out a task. Thus, it is creative to use new techniques in system development and to combine different kinds of technology to develop new products. It is, however, also creative to use well-known techniques and technology in new ways, whether be it creating a better process or utilizing technology in a way better suited to the application area.

With this notion of creativity we may include the creative aspects of everyday system development. A broader understanding of creativity therefore makes a basis for utilizing the large body of Scandinavian system development research from the last fifteen years in

[11]Torgersen (1972) defines a profession as a relation between an occupation and a specific education. Cf. also Freidson (1970).

everyday system development, by indicating that creativity not only takes place in research. At the same time this notion of creativity can be used as a means for legitimating system development research and as an argument against system developers becoming superfluous. In this way creativity legitimates both education and research about system development.

Summarizing the discussion we feel that the concept of creativity can be utilized for marketing the Scandinavian tradition of research on system development. As the "narrow" definition of creativity does not fit the tradition it is supposed to market, we find the discussions of creativity in system development quite meaningless. Thus, we do not see that the discussions about creativity contribute to new knowledge about system development, neither for the people inside nor for the people outside the Scandinavian research tradition of system development.

References.

Bachrach, Peter & Morton Baratz (1962): *Two faces of power* in **Americal Political Science Review** Vol. 56, No 4

Bansler, Jørgen (1987): **Systemudvikling - teori og historie i skandinavisk perspektiv** ("System development - theory and history in a Scandinavian perspective", in Danish), Studentlitteratur, Lund

Bjerknes, Gro et al (1987): **Læring som forutsetning for design** ("Learning as a prerequisite for design", in Norwegain), Report no. 2 from the Florence project, Department of Informatics, University of Oslo

Bjerknes, Gro & Tone Bratteteig (1987a): **Å implementere en idé - samarbeid og konstruksjon i Florence-prosjektet** ("Implementing an idea - cooperation and construction in the Florence project", in Norwegian) Report no. 3 from the Florence project, Department of Informatics, University of Oslo

Bjerknes, Gro & Tone Bratteteig (1987b): *Perspectives on description tools and techniques in system development* in Docherty et al (eds.): **System Design for Human Development and Productivity: Participation and Beyond** North-Holland, Amsterdam

Bjerknes, Gro & Tone Bratteteig (1988): *Computers - Utensils or Epaulets? The Application Perspective Revisited* in **AI and Society** Vol. 2 No. 3

Budde, Reinhard et al. (1989): **Prototyping - An Approach to Evolutionary Systems Development** Springer Verlag, Berlin (forthcoming)

CSCW (1988): **Proceedings of the Conference on Computer-Supported Cooperative Work** Portland, Oregon, Sept. 26-28 1988

Ehn, Pelle (1989): *Playing in Reality* in this volume

Ehn, Pelle & Morten Kyng (1987): *The Collective Resource Approach to Systems Design* in Bjerknes, Ehn & Kyng (eds.): **Computers and Democracy** Avebury, Aldershot

Freidson, Eliot (1970): **Profession of Medicine** Harper & Row, New York

Kensing, Finn (1987): *Generation of Visions in Systems Development: A Supplement to the Tool Box* in Docherty et al (eds.): **System Design for Human Development and Productivity: Participation and Beyond** North-Holland, Amsterdam

Munk-Madsen, Andreas & Berit Thaysen (1989): *Conditions for Creativity in System Development* in this volume

Torgersen, Ulf (1972): **Profesjonssosiologi** ("The sociology of professions", in Norwegian) Universitetsforlaget, Oslo

Wittgenstein, Ludwig (1967): **Philosophical Investigations** Basil Blackwell, Oxford

'THE SAME PROCEDURES AS LAST TIME' when developing and implementing a new computerized information system?

© *Bengt Brattgård, Kjell-Åke Holmberg and Gunhild Sandström*
The AMIS group
Applied Psychology, University of Lund
Information and Computer Science, University of Lund

Abstract: As common bases for this paper are different parties´ opinions within the same company of how computerized data and information technology are used today together with assessments on earlier development works. Our paper therefore shows three different stories about a system development project in a non-creative environment for designers and users. Our study is done in a big manufacturing company.

We present a case which offered limited possibilities for creativity in the process of systems development. The pleased and displeased project workers and system users from different positions in the company retold their experiences of the systems development process and results. We especially expose the division of power forms, the business agreements, the project organization and the working forms, which hindered creativity when developing the information system. Some peculiar and hasty made decisions and solutions became the result of the development. The case is analyzed from the viewpoint of knowledge, autonomy, responsibility and how these, or the lack of these affected motivation and creativity for all concerned.

The case study is the first step to forming new development habits and our paper will conclude with a discussion on ways to break bad developement strategies.

The AMIS group: Information and Computer Sciences & Applied Psychology
University of Lund
Sölvegatan 14 a, S-223 62 LUND, Sweden
telephone 46 46 107000

'The same procedures as last time' when developing and implementing a new computerized information system?

This research is about a case study within a 3-year project "Forms for Responsibility and Cooperation in Continuous Systems Development" (AMIS[1]) supported by MDA[2]. The stated goal of the entire project is to induce the building or rebuilding of information systems which offer more variety of work content and a more suitable work organization for everybody concerned. In practice, we study and hope to improve systems use and development and its environment. As practical results we expect improvements in quality of work design and efficiency for end-users, for information mediators and for systems designers, as
* job satisfaction, personal development and self actualization
* higher skilled information workers
* better adoptions of results and processes with various tasks
* more interchange between the information work and the changing milieu
* possibilities to improve service to customers and third parties.

Our theoretical aim is to test and improve new systems development models. To that end we have to learn on the basis of real situations. We intend to find out what characterizes information systems use and development with user involvment in certain organizations. At present eight researchers from Information and Computer Science and Applied Psychology from University of Lund work in a research team in cooperation with the employees in these organizations. Our goals and starting points and models are more carefully described in Nissen, Sandström & Ekvall (1988)[1] and Nissen et al. (1989)[2]

The story told by minutes and internal documents

The process of designing a new data system for manufacturing control and material planning started during 1984 with a requirements specification made by an external consultant. This investigation was based upon user viewpoints collected during a series of interviews. The consultant's report suggested a technical design with a main goal for capital rationalization of assets tied up in raw material and stocks of finished goods.

At the same time a new computer and data system had been installed at the economics department, and the management decided to throw out the existing computer used for manufacturing

1. In Swedish the acronym is AMIS. It stands for Ansvars- och Medverkansformer I kontinuerlig Systemutveckling.
2. MDA stands for Människor - Datateknik - Arbetsliv (Humans - Computer Technology - Working Life). It is a multidisciplinary research programme jointly funded by the Swedish National Board for Technical Development and the Swedish Work Environment Fund.

planning. It was also stated that any new data system should be able to communicate with the economic subsystem to form an integrated whole.

In January 1985 a startup meeting was held in the implementation phase. At this point of time they had only a year up till the switchover from the existing system should be completed. During this meeting it was decided to form a steering committé, a user group and a work group, where the user group reported to the steering committee.

The future solution was discussed, and during this meeting one of the users expressed the fear that they had to adopt their work organization to a technical solution. A German system, that had been ruled out previously for a German part of the company due to great differences in production structure, was now discussed. One of the management representatives stated that he had been talking to a representative from another branch of the company, and claimed that there were solutions how to handle the differences in production structure. He demanded that the group should go to Germany to study how the problems could be solved.

The external consultant, who had been one of the two members of the working group, did not show up in any minutes after February, and we were told that he had been trown out. During february it was decided to accept the offer from the computer company, that had delivered the computer system for the economic department's subsystem. It was also decided to choose the German solution, and the computer company had agreed to buy the system from the German firm. There is no document in the company records we were allowed to look into, that shows whose decision it was to choose this vendor. The only document, that gives any clue to where the decision was made, is a letter of intent from a management representative.

The contract with the computer states that the system should be up and running in October. The contract also states that there should be three main groups in the project.

- A system group with members from the computer company and one representative from the client organization. Their responsibilities should be to implement changes and install the system. They were also responsible for the training of the other project groups.
- A user group consisting of members from departments involved and the project leader from the computer company. Their responsibility was to develop user training and manuals and act as a reference for specification provided by the other groups.
- A steering committee consisting of managers and middle managers from the company. Their responsibilities according to the contract were to define and implement interfaces, routines, and conversion routines.

At the end of March an information meeting was held for the personnel. At this information meeting the project organization and the future system was presented.

The first minutes from the steering committee in the available company records dates from June. At this meeting it was decided that the specifications from the computer company must be signed and formally agreed upon by one member of this committé. It was also reported that so far the time table was held by the system group. A claim from the user group for more resources was discussed.

During September/October meetings in the steering committee all groups stated that the deadlines should be kept. Of special interest is a fear of a delay and a request for an alternative implementation plan expressed in a memorandum by one of the user group members. The steering committee decided that an alternative plan should be laid out and that one of the committee members should inform this user group member of the "perfect" status of the project.

During September there were indications of hardware and software problems. Among other things the important bill-of-material processor loaded down the computer heavily, and could not be run for any reasonable amount of time. Still the system group reported that there were no problems with the deadline, even if there was a slight problem with the size of the computer system, e g there was reported a serious fragmentation problem with the disk.

In the beginning of October the steering committee was made aware of the fact that the deadlines probably could not be kept, and that a catastrophe plan had to be worked out. Despite the problems the manufacturing planning was laid over to the new system at the agreed date - 15th October.

The minutes from the meetings in the steering committé shows that major parts of the new system worked very unsatisfactorily, and that the manufacturing planning in the company had to work on an ad hoc basis with a minimal support from the new system. The minutes also show that some vital functions still has to be run with the older system.

There are demands from the user group for support from the computer company with the user training, that is not met.

In May when the manufacturing planning and manufacturing still is not working, a catastrophe meeting is held and a formal letter of complaint is sent to one of the vice presidents of the company and the board.

Finally in autumn 1986 (with a delay of almost a year) the system works, but still far from satisfactory. In May one member of the steering committé held an interview with some of the users. From this meeting the following citations are representative for the feelings.

"Response times too long > 1 minute"
"Buy boxing balls"
"The training was no good"
"No manuals"
"The routines are 18th century — The possibilities are 21st century"
"Too much time at the terminal — I believe the system is good"

The story told by the users

The users did experience some shortcomings with the old system in that it was complicated to use and that some calculations could not be done in systems. The initiative to change the system did not come from the users. They experienced the initiative to come from the economic department. The work of finding a system to the computer started when a new official arrived at the administrative unit.

A work group was formed to examine suggestions for alternative systems. Representatives for the users joined the group after a while. Groups for the management and steering of the project were also formed. The working group was in Germany and studied a system for the brand of computer the economic department had chosen. In an early stage the work group showed specific interest in one system but then nothing happened until the supplier decided to market the aforemantioned system in Sweden. In the early stages the company hired a consultant who was good and knowledgeable. The consultant was thrown out when the costs rose.

They wanted the new system to be easier, to build a data-base and to do some further calculations. Most of the desired functions already existed in the old system. They were skeptic to the fact that the new system based the schedules on orders from the customers as the company did not plan their production in this way.

When the contract for the old computer system was cancelled there was no time to adjust the faults that were detected in the new system. They were so many that the company considered taking back the old system. Production was maintained with the help of old printouts. A decrease in orders also helped so stock never ran out. The management did not listen when the users commented on the shortcomings of the new system. They meant that "The responsibility was delegated, but not the authority". It was not clear who would pay the costs for the necessary changes. In the beginning the lead times were long but after the change to a more powerful computer they got shorter.

One of the users representatives had the responsibility to train 10-15 persons. The staff was trained on the new system but it did not work in correctly. The education was thereby not the best. The representative who worked with the training and education stated that it was tough . There were no manuals. The company still suffers from the lack of basic training, as there has been no opportunity to give any further training.

The users representatives are on the whole quite satisfied with the system as it works today. One part of the company is maybe more dissatisfied than the other even though the manager of the project was in this company. Maybe they do not use the system in an accurate way? In the future there are plans to incorporate subsidiaries in the system to achieve gains in coordination when dealing with suppliers.

Today it would not be possible to handle a new change of system but it would be advantageous if the search for the system of 1995 is started today. One idea from a user in the enterprise is a network of PCs connected with a data-base. Both technology and responsibility then is delegated, and it is possible to use both the intelligence of the computer and the intelligence of humans. A longer period for systems implementation and introduction is a necessity. The users should be able to concentrate their efforts on the implementation only and be releaved from ordinary work during this period. The later you get involved in the implementation of a new system the greater the problems are later on. Now more than three years afterwards their users claim that it is only recently they can handle the system in a satisfactory way.

The story told by the managers

This description is based on interviews with two managers who participated in all the phases of change of the information system. They were strongly involved in the whole process and remarked that they well remembered what happened.

Twelve years ago there was no computerized control system for administration in this company. A decision was made to implement such a system. This system, however, had extensive limitations. The system was non-interactive, only distributed numerous computer lists with commands. The files had to be concluded every month which meant that the people had to start from scratch again. It was a very hard and tough system that very few people understood.

Employees who could mangage the system had difficulties in explaining it for other people. They could not even understand why other people had difficulty with working with the system. The system was insufficient. It was unwieldy and heavy and yet more people were to be connected to it.

Six years ago the company employed a new general manager and installed a computer of a new brand. The company started to look for a new data system in order to replace the old hard one. A consultant collected the requirements from the users and pointed out economic savings and goals which could be reached. With support from a new computer-based system, the management wanted to establish a reduction in the capital tied up in 'half fabricated products', a reduction of stocks and their customers debts. They also wanted to do something about the lack of components for manufacturing. A year later a member of management told the others what should be improved by the new system. It would be possible to directly connect with display units. With an interactive system it should also be possible to increase efficiency while at the same time work should be made more enjoyable. The administrative rationalization would imply that the users would have more time for more important activities. From the beginning there was a great optimism among the intended users and these had visions. They wanted such a system.

One of the managers comments about what happened. He meant that there should have been more time and effort for creative solutions and. At that time most of the creativity resources have been allocated to product developing and marketing. The employed consultant was later taken away by the management without asking or telling the personnel involved. He thought that there should be a better decision basis before applying a consultant.

A system of the same brand as the computer was chosen. It was a system which could also be used abroad. The main reason for the choice of the system was that this system was the cheapest, they said. After this decision two very active groups were established, a control group and a work group. The two leaders joined the control group. The chairman in this group was the general manager. In this group were also three representatives from the company who manufactured the computer and the system. They met every other week. The work consisted of the consultant and two user representatives. They were given the responsibility for education. They had to educate the rest of the personnel. This education took a lot of time in the departments and everybody felt as if it was a critical state and it was very tiresome.

The managers had expected more help from the manufacturer of the computer and system. When this company received their money they lost interest. They should have prepared the users for the large amount of work that was at hand and the users should have listened more and taken more advantage of the knowledge that actually was to be found. The end users were unable to influence the design. It was true that some of them went abroad to look at various layouts of displays and these were understandable for the users. However the algorithms and the calculations behind what was shown could not be understood at all. The managers meant that the personnel in the departments should have been responsible for the development. It was forgotten for example to write an instruction manual for use in one of the departments. Later when the people who could handle the system left the firm, the situation for the rest became a disaster.

On D-day the new system was started without parallel running of the old system. All energy was to be put in the new system. Knowledge was now orally transmitted from the responsible people. This was unsatisfactory as it was forgotten after a short time. It took almost a month before they were sure that the new system would function as expected. They hade to run on Saturdays and Sundays. Nobody trusted the system. People remarked that they felt that they were in a trap with long lead times and other troubles. They were forced to start the old heavy system again in order to get a better base from which to make decisions. The result therefore was that they ran one heavy system parallel with an unreliable system. Moreover, the implementation became more difficult as the documentation of the earlier system was weak.

Three years ago the enterprise was divided into two separate companies and the two managers became employees in different enterprises. Both the companies were to use the new system, in spite of the fact that the system was built for activities of one of the enterprises. It was found out earlier that there were different kinds of requirements in the two companies, but when the system was run difficulties arose in both. These were extremely great in one of the firms. It really was a crisis according to these leaders.

It was decided that the computer company and the consultant must change the system, which had to short a perspective in the algorithms. The long run work was totally out of it. One of the managers was angry with the general manager, who should had seen to it that the computer department of their own company became responsible and set about the problems. The new system made it very difficult to furnish the customers with goods according to the set delivery time. As time went by the computer company suggested changes and made alterations. Few improvements were noticed after that. The enterprise changed its general manager.

However the problems had not come to an end. A new version of the display units was installed, but the response times then became so long that it lead to chaos and irritation. It was discovered that the machine capacity was insufficient. The waiting times, including the response times, were unacceptable. In the beginning the employees tried to examine several challenging situations of this kind, but as time went by they completely lose their spirit. "The system was meant for year 2000 and the routines were from the 18th century, and the education was inferior". That was the common meaning among the personnel. A new larger machine of the same brand was installed and for a large amount of money two people were sent on renewed education at the computer company but to no avail. The practical and theoretical education that was needed was not available. One of the enterprises employed a new leader.

The two enterprises were treated differently. There was a concentration on one of the companies, but it did not help due to the lack of knowledge, and the other company was more or less neglected. Education was started again, this time internally through two old users together with a new employee from the computer department, which yielded results. It was possible eventually to manage the system and to utilize its potential. The computer company however wanted the organization to buy a new system.

The trade-union has only been informed once at a meeting four years previously on the manager´s intiative.

To sum up the implementation was too sudden and the education was too bad. One of the leaders felt accessory to this. The computing department has changed personnel since then at 100 per cents. Nobody remember how much the system has cost or how much time that could have been spent on its development and implementation. One of the leaders had himself put one or two man years on the system implementation. The implementation problems have lead to unpleasant results, that the mangagement higher up have not been able to accept. Now some positive results have been able to be discerned. People have started to handle the parameters. At last they dared to utter the concept "capital rationalization". The goal that was put by the consultant from the early beginning was now reached.

But the system is still patched and mended. New errors and not yet met expectations are detected. During the last six months it was discovered for example that old orders stayed in the system in spite of that they should not. Sometimes certain functions which should be in the system are missing. "The organization was the first one in Sweden that got this particular system and because of that the implementation was not quite painless." This was said in order to defend or explain the behavior of the computer and system company. "But if an investigation was to be done today the result certainly should be a decision somewhere to introduce a similar system again," the managers convincingly told us. However, there has been a great deal of talking about buing a new better, different, smaller and more powerful system in one of the enterprises. Perhaps, it may not be a repeated story - the next possible system development?

Explanations and analyses

The impact of business agreements and economical/technical constraints

The five most critical factors in the systems development project are

- The time factor
- The company computer policy and the desirable integration of the studied system with the data systems of other departments into an integrated whole
- The choice of a system package, where the users had no opportunity to influence the choice
- The structure of the project organization stated by the business agreement with the computer vendor
- The vendor's lacking know how of manufacturing planning

The ultimate limiting factor seems to be time. A data system should be designed, tested and implemented and a computer system to go with it should be chosen and installed in less than 9 months. The short time period depended upon the desire to get rid of the cost for the old system by the end of 1985, when the current contract period was at an end. Together with a feeling for the complexity in a manufacturing planning system this probably automatically lead to the need to look for a package. Economic reasons and the short time period also forced the management decision to choose a single contractor for both hardware and software. A natural consequence of this was that the computer specialists were the real and only leaders and designers in the project.

The desire to integrate this new system into a companywide system then probably made the the chosen system the only solution available. The only remaining design decisions then were how to decide upon minor changes to the system's interface and how to restructure existing planning activities.

Going from a requirements specification, where at least the user's know how and creativity had influenced the design, to a design more or less cut in stone reduced the employees to errand boys for the computer specialists. This division of responsibilities in fact was laid out for them by the project organization stated in the contract, where the project organization was derived from a ”standard” solution in the computer field. There doesn't seem to have been any regards to the current situation and any wish to take care of know how other than purely technical for the adaption of the data system to the planning situation.

With the chosen system this division of responsibilities proved to be highly unsuccessful, as the knowledge of manufacturing planning was with the user. We have here a situation, where the work organization completely had to adapt to a technical design, and where the designers neither had time to listen to user critique nor had any competence within the application field. Together with complex software problems and a computer system with insufficient capacity, the system group completely concentrated their efforts to the software design issues.

This seems to be an all to common situation, where a short time period, standard procedures of the computer field, and an unfinished package forces the computer vendor into an impossible situation they should have anticipated, and the company into a bad system. The situation can be regarded as a "Tell and Sell" situation, where real user influence and creativity is reduced to a minimum.

Project organization and responsibility questions

Project organizing and responsibility when dealing with a turbulent and undpredictable environment are not easy to decide about. Therefore, paradoxically, it is too often decided to organize and respond almost as it was done the last time in spite of the fact that it has led to the actual situation at the end.

A project is featured to be limited in time and space. A project is established for specific aims. In this case the aim was to develop a systems launch. For this purpose three groups with partly different goals were established: a reference group, a control group and a work group. The groups were not established to represent all the interests in the organization. These interest parties, however, were not quite easy to detect at the time when the groups were established. The groups had different kinds of tasks. The reference group had the power, through money, and decided about the choice of computer system brand. The control group had implicitly the task from persons in the reference group and computer selling company to control something. The work group had explicitly the task to design the information (or maybe just the data) system.

None of the members of any of the groups seem to have contributed with any new good ideas. The reference group contributed with decisions and the work group members anyhow did the design, the same as last time there was a systems development. Nothing really new and helpful was made.

In a systems development one could speak about four different kinds of responsibility:

- responsibility for the job in question
- responsibility for the development of an information system to be used in this job
- responsibility for the 'ready-made' information system
- responsibility for the use and the continual development of the information system.

Regarding the responsibility for the job, it is often so that the formal responsibility lies with somebody else than the person that should do the work. This somebody is often a foreman or a manager of some kind. He (it is often a he) will be honoured when the job goes well. The person who work will feel psychologically bad, as the responsible person lay over the implicit complaints with the silent argument 'not safisfactorily done work'.

The responsibility for systems development lies with a person or with a group of persons who have been pointed out to see to it that the new information technology is on place in time and that it functions technically. For the use in milieu respond the working people and sometimes also their managers.

The ready-made data system will become an object, which often will be the alone headache for a computer department or for the department that are going to use the system, not as we would wish, a concern for both parties.

The further development of an information system is hardly initiated by the end users. Initiatives emanate from the computer department, where people mean that "now, there are more modern technical stuff to utilize - let us do so". Initiatives may also arrive from a department for Administrative Development with resources and possibilities to distribute these on well-argued project. Decisions in such questions are then based upon the managers arguments for information systems development at their departments. These arguments have very different foundations and strengths for the needs by the workers at the terminals.

Taken together the different phases of responsibility concerning systems development and use look as in the following table:

	Formal responsibility	**Real responsibility** **'They who do the work'**
Work with computer support	the manager, the foreman	white and blue collar workers
Systems devlopment	special project leader	task groups
Readymade data system	computing manager	operators, programmers
Systems use	nobody	white and blue collar workers
Further development	computing manager or department for adminstrative development	task groups

Table 1: Distribution of responsibility concerning the systems design (according to G. Sandström)

Very often, not even the people with the formal responsibility have the power, the resources or the permission to do or to order to do something about smaller not satisfactory work situations.

Creativity and Innovation In the Process of Implementation

We do not intend to discuss the organization as whole in terms of creativity and innovation but the implementation of the new information system. Nevertheless it is important to regard the role of and the conditions for the implementation in the organization. It is not meaningful to discuss the the best solution of system and implementation separate from its function in the organization. The work of the system designer is by necessity bound to the function of the system and to the people using it. Means-end should be to offer jobs to people with opportunities to use and develop their skills in the way they want to. To have a good process of implementation it is necessary to have some conditions.

Structural conditions:

You can not tell how the creative organization looks à priori. It differs widely according to time, environment, the task for the organization and people involved but they seem to have certain characteristics in common. These are division of power, openness and flexibility. Looked upon from a traditional bureaucratic position the creative organization seems to be complex and confused.

According to Ekvall (1988)[3] the organization consists of three systems, one for ideas and development, one for decisionmaking and one for performance. To be able to have control over the own work one should have access to all three systems. I traditional organizations the distribution of employees on the different systems are strictly regulated. It is common to organize the development in a sub-organization for research and development where the necessary degree of freedom can be obtained. In this way it is still possible to control the production and the decision-making. The result is however often that the people responsible for performance have no way to influence the development of the organization and the people responsible for development may get freedom enough within the group but have no exchange with other parts of the organization. To broaden the possibilities to their own work situations admittance to the idea system is necessary.

Group conditions:

Social pressure is in some literature viewed as a negative factor for creativity. It is said to hold back the individual not to do that extra an unexpected. There are some effects of social pressure and it is worth discussing the influence of e g belonging to a collective of workers.

Another problem is that people with various backgrounds are prepared to participate in the creative process in different ways. A method to deal with this problems are described by Gustavsen[4] as shown below.

Problem solving:

Argyris, Putman and McLain Smith (1987)[5] refers to Schön's distinction between problem solving and problem setting. Problem setting includes the process of "naming and framing" what is to be solved and is in traditional organizations taken care of by the decisionmaking system. In this way the management often decides what will count as a solution. The tasks for the developer are mainly to solve the managements problems and to improve existing products.

Even if the employee today often is encouraged to present ideas and improvements there are often no channels for him to be heard except for the suggestion box. The free exchange of information and ideas does not exist and the few suggestions more often deals with work environment than the work itself. It is obvious in this case that the problem was seet on top mangaement level whil middle managers had to solve it supported by the staff personnel.

The company studied was open and flexible compared to other Swedish companies of the same size. During the implementation period there was, as mentioned before, a shift in ownership and a decreasing profit thus followed by this and other changes in the organization. This made the conditions uncertain and may have increased the time pressure mentioned above.

Problem solving in groups:

As the interest the AMIS-group lies on continual or evolutionary systems development there certainly are more than one person that ought to be creative in the development. Gustavsen describes a model for 'Democratic Dialogue'. To be called democratic o process has to offer everybody concerned an opportunity to participate, he claims. The participation may depend on the local condition but this criteria is basic. Another core element in 'Democratic Dialogue' is that the participants in this process has to be trained in this way of decisionmaking. This training is handled in groups of different compositions with no authorized trainer.

As we did not follow the work of the groups ourselves it is not possible to discuss the work within the groups in a comprehensive way. In spite of this it is possible to draw the conclusion that the composition of the groups and the cooperation between the groups were not the best and most certainly influenced negatively on the result. The role of each group differ over time and was therefore unclear.

Starting points for new forms of systems development

With the above reflections one could ask if it is possible to build a better project culture into the base organization in order to take care of new ideas, to make good and more purposeful decisions about systems development and to work out the decisions through learning and responding to the changing environment. Support for this reasoning we also have in a hypothesis by Burell & Morgan who a. a. argue for a "strategic management which foster the ability of the organizational unit to learn and respond to the environment" as a need for an organization and its organization units in order to handle turbulence and unpredictable situations with regard to the mode of research and development. (Burell & Morgan, 1979 pp. 177 & 178)[6]

* There has to be a method to introduce information systems work by self-managing teams in order to take care of abilities and advantages of the workers innovative and decision-making capacity. This is our first key-point.

* The impact of business agreements is very strong in the sense that the deadlines must be kept at any cost - such as bad systems design and dissatisfaction and pressure among the personnel. The business agreements are very weak in the sense that the capacity and the functions of a computer system are not understood by future users, and even in many cases important system functions never work at all. Agreements should be kept and they should be understandable to oneself and ones field of responsibility. There must be no pretending or just believing comprehension. This is our second key-point.

* With information technology as working support the responsibility for an information system in use and for its continuous development should be shared between the computer personnel and the work organization. It is a common concern. A continuous systems development deals with development of environments and techniques, where both users and experts of data system are able to contribute with their competences. This is our third key-point.

* Moreover the power, formal responsibility and real responsibility - that is the work itself - are not united. The power is with leaders and the work tasks are with the white and blue collar workers. However, the formal responsibility is with middle managers either connected to the work tasks nor with the power to change them. More effort should be to unite power, responsibility and work in the same persons - the workers and the clerks. In order to make possible continuous systems development of good quality there must exist abilities to enable enthusiasm, creativity, initialization, learning and action in work. Otherwise, the developed information system may lead to declined efficiency and dull work. The resources have to be available for all parties involved. The working people should thus get more power, in form of money and knowledge, in order to initiate and work out their own suggestions for change. This concerns, of course, also system designers. This is our fourth key-point.

[1] Nissen H-E, Sandström G & Ekvall G (1988) "Ansvars- och Medverkansformer I kontinuerlig Systemutveckling - teoriansatser för ett projekt" research report LU-ADB-R:88-3, Information and Computer Science, University of Lund, Lund, Sweden (in Swedish)

[2] Nissen H-E, Brattgård B, Ekvall G & Sandström G (1989) "Gemensamma utgångspunkter och -modeller i AMIS-projektet", research report LU-ADB-R:89-3, Information and Computer Science, University of Lund, Lund, Sweden (in Swedish)

[3] Ekvall G (1988) "Förnyelse och friktion" Natur och Kultur, Stockholm

[4] Gustavsen B "Creating Broad Change in Working Life. The LOM-Programme." Quality of Working Life Center, Ontario

[5] Argyris C, Putman R & McLain Smith D (1987) "Action Science", Jossey-Bass, San Francisco

[6] Burell G & Morgan G (1979) "Sociological Paradigms and Organizational Analysis" Heinemann

Acquiring System Development Knowledge by means of Qualitative Interviewing

Lars Bækgaard and Carsten Sørensen
Institute of Electronic Systems
Aalborg University
Strandvejen 19
DK-9000 Aalborg
Denmark

October 13, 1989

Abstract

Empirical researchers are in need of a broad spectrum of methods in order to understand the complex problems surfacing during development and use of information systems. The aim of this paper is to discuss the idea of qualitative interviewing as a valuable part of the system development researchers tool-box. We describe our use of qualitative interviewing in a case study concerning development and use of a computer-based purchase support system. First we present some of our interview results in order to illustrate the potentials of qualitative interviewing. We then communicate our own experiences by means of a rather detailed description of our research activities. In order to clarify the major points we give a short summary of the content of these activities.

1 Introduction

In this article we present our use of qualitative interviewing.[1]. We describe our experiences in order to illustrate how we used qualitative interviewing to acquire valuable system development knowledge.[2]

We used interviewing as the primary research method in combination with additional activities such as observation, document reading and informal talks. Our idea was to let problem settings from a specific system development project be inspirational backgrund for our reflections. The interviews enabled us to make descriptions of a system development case and to suggest new elements of system development methods and theories (Bækgaard 1988, Sørensen 1989a). We find that empirical research is necessary in order

[1]Our experiences origin from an empirical research project (Bækgaard & Sørensen 1988)

[2]The use of qualitative evaluation methods in general is described in (Patton 1980) and differences between quantitative and qualitative methods is discussed in (Bryman 1988).

to make stronger connections between methods/theories and practice and in fact many researchers have been using empirical methods such as interviewing and observation with the purpose of gaining knowledge about system development.[3]

In (Kasanen & Suomi 1987, p.324) it is stated that because of its young age system development research is plagued with lack of established research methods. Consequently there is a need for research on methods, techniques and tools that can be used to support empirical system development research. We would like this paper to be read as a description of one among many possible methods for empirical system development research.[4]

In section 2 we present three examples of results from our interviews. We argue that the use of qualitative interviewing as research method brought about these results. Section 3 presents our use of qualitative interviewing in the project. We use a fairly large proportion of the article to do this because we find that a presentation of an example is more instructive than an abstract presentation of the method. Section 4 concludes the article.

We would never claim that qualitative interviewing is the answer to every research problem. On the contrary we recommend the use of as many complementary research methods as possible and in fact we supplemented our own interviews with observation, informal talks and document reading. But we do claim that qualitative interviewing has some strengths that makes it a valuable part of the system developers tool-box. It is our hope that we will be able to convince the readers that qualitative interviewing has powerful potentials.

The reader should note that this version of the paper is not the final one and that it is devoted to presentations of our use of qualitative interviewing. In a forthcoming version we will add reflections on some of the important choices related to the use of qualitative interviewing and reflections on the relations between qualitative interviewing and other types of research methods.

2 Was it worth the effort ?

Although the subject of this paper is our experiences in research based on qualitative interviewing we have found it important to discuss some of our results briefly before we turn to the discussion of methodical aspects. Firstly, we believe that it will make it easier and more interesting to read about our experiences. Secondly, we find that an argument in favor of a research method must include an argument for the possible quality of the results.

The purpose of our interviews was to evaluate development and use of a computer-based information system in the company T-Purchase. One system developer and three users

[3]Examples can be found in (Wynn 1979), (Andersen et.al. 1986), (Holmquist 1989) and (Ehn & Kyng 1985) using observation and (Borum & Enderud 1981) and (Andersen & Madsen 1988) using interviewing.

[4]Other examples are (Mumford et.al. 1985) on empirical methods, (Jepsen et.al. 1986) on diaries and (Etzerodt & Madsen 1985) on basic choices when designing research.

participated. The management in T-Purchase had decided to replace their self-made purchase support system by a pre-made software package. The replacement project was terminated and the new system had been used for some time when our research took place. Thus there was no really alternative to interviewing if we wanted to examine the project and the use activities.

The immediate result of a qualitative interview is a collection of quotations reflecting what was said during the interview and in section 3 we will discuss a range of possibilities for the use of such quotations. For the purpose of illustrating our results we have selected three examples that are very close reformulations of quotations. A more detailed discussion of these and many other results can be found in (Bækgaard 1988, Sørensen 1989a)

The past: The users wrote their own manuals based on experiences from problem situations. The prewritten manuals did not contain any useful information that could help the users to solve their problems. Instead they developed a practise of note-taking which turned out to be useful in two ways. Firstly, the mere writing improved their problem solving abilities. Secondly, they used the notes if a problem occurred more than one time.

The present: The system developer worked in the same physical and organizational environment as the users. This arrangement had a positive effect on the users perception of the quality of the new system. In problem situations it was feasible for the users to ask the system developer for help and this enabled them to solve a lot of their problems quite fast. The consequence was that the negative properties of the new system did not irritate the users very much.

The future: The system developer intended to base future redevelopment of the system on use experiences. The users had a lot of change requests but the system developer minimized the number of changes that were implemented before the old system was replaced by the new system. He was convinced that use experiences would show that only some of the change requests were actually relevant.

These result examples illustrate two important aspects of qualitative interviewing. Firstly, the interviews allowed us to "look" into the past, the present and the future. The use of interviewing allowed us in an efficient way to evaluate the participants perceptions of what had happened, what did happen and what should/would happen. Secondly, the qualitative characteristic of our interviews allowed us to obtain surprising results. In fact none of the results above was foreseen in our questions. They came about because the interviewees found them important.

3 An example

The purpose of this section is to illustrate some of the potentials of qualitative interviewing. We do this by means of a description of our own experiences in qualitative interviewing stemming from a system development research project.[5] The following six subsections each cover a major activity in the research project i.e: (1) Preliminary research design;

[5]The project and its results are documented in (Bækgaard & Sørensen 1988).

(2) Acquiring situational knowledge; (3) Designing questionnaires; (4) Doing interviews; (5) Documenting data; (6) Using and interpreting data.

3.1 Preliminary research design

The starting point of our research project was that we wanted to study development and use of computer systems in real world organizations. More specifically we wanted to study relationships between development and use and the properties related to these two types of activities. Before we initiated the actual research work we made arrangements with two organizations in which we could make case studies, we decided to use qualitative interviewing as our basic research approach and we made an overall plan for the rest of the research project. Let us take a look at these three initial activities in turn.

Our attempts to establish contacts to candidate organizations turned out to be a rather time-consuming activity and the limited number of relevant and available organizations constrained our research possibilities. We contacted eight organizations by means of letters in which we shortly described our research ideas, our expectations to participant organizations use of resources and what they could expect in return for their effort.

In some organizations we did not mail to the right person and some letters circulated inside the organizations in one or two weeks. Some organizations were relevant for our study but they did not have sufficient time and people to participate in our project. Other organizations would like to participate but they did not seem to have any interesting use activities or development projects to study. One organization was anxious to participate because of bad experiences with other research projects. The result was that only two organizations was relevant and available and we spent more than one month before we had made arrangements with them. Although this was not full time work it delayed our research project. In this paper we have only described our research experiences from the order-producing company T-Purchase because the experiences from the other organization is very similar.

For a number of reasons we decided to base our research activities on qualitative interviewing. We did not have enough time to participate in or to make direct observations of development and use activities so we had to find a way to study the past. Instead of testing specific hypotheses we wanted to learn from the specific situations in T-Purchase. In the previous section we have presented examples of the results of such learning that would not have been possible if we had based our research solely on hypotheses. We found that qualitative interviewing would be the best way to handle these conditions and our experiences have convinced us that it was a good choice.

The last thing that we did before we initiated the actual research work was to make an overall plan of this work. We decided to supplement the interviews by means of informal talks, document reading and system demonstration. The following subsection contains a discussion of the purpose and content of these activities. We decided to do four interviews - one with a system developer and three with different kinds of users. Originally we had planned to make only one user interview but the system developer told us that there were three kinds of users who worked with the system in rather different ways.

3.2 Acquiring situational knowledge

Before we prepared and did the interviews we acquired knowledge about the system development process in T-Purchase, the purchase system and the organizational setting. We did this in order to be able to design questionnaires that directly reflected some of the characteristics of the situation in focus and to interpret the interview results. We use the term situational knowledge to denote this kind of knowledge as opposed to general knowledge about computers, organizations, human beings and so on. We used three different approaches to knowledge acquisition which turned out to enhance the quality of other research activities.

Firstly, we had informal talks with the system developer who told us about the standard system, use situations and the system development process. He gave us a background for understanding why T-Purchase replaced the existing system with a standard system. Before the transition T-Purchase had several systems each supporting a specific function in the company. It was not easy to integrate the systems and let them share common data. The chosen standard system was actually a subsystem in a larger system based on one database for the whole company. Besides the system developer expected to get rid of some of the maintenance problems T-Purchase had with the old system.

Secondly, we read documentation including technical system descriptions, project plans and notes from project coordination meetings. The reading of project plans and notes from meetings was especially useful. It gave us a rather detailed picture of the major events and decisions in the development project and we compared visions captured in the plans with the reality captured in the notes from meetings. We used this knowledge later on when we prepared the interviews by means of questionnaires and we used it when we used and interpreted the interview data. Unfortunately the system descriptions was not of much help to us concerning information about system properties, but provided important knowledge about the decision process.

Thirdly, we participated in a system demonstration focusing on some of the central system functions. We saw the most important screens and the system developer gave us a guided tour through parts of the system focusing on its relations to work processes. This gave us a background for understanding when the interviewees talked about specific system properties and their work with the system.

3.3 Designing questionnaires

After acquiring situational knowledge we prepared the interviews further by making questionnaires. Figure 1 shows an example from one of our questionnaires (Bækgaard & Sørensen 1988, p. 32-33). For each topic the questionnaire was divided into three parts. The first one was an *italic* part expressing our intentions. It was meant to be a reminder for the one of us who were going to be interviewer and we did not tell the interviewees about the existence of this part. The second part was an introductory piece of text to be read to the interviewees in order to prepare their minds to the topic in focus. The third part was a sequence of carefully worded questions to be read to the interviewees. Some of the questions was augmented with *italic* probes to be used if the interviewees did not give a comprehensive answer.

The purpose of the following questions is to encourage the user to tell about his experiences in relation to the process of learning to use the new system. We want to explore the initial training and learning in work situations where unexpected events can occur.

The following questions is about your experiences in relation to learning to use the new system.

How did you learn to use the system? *(experiments, asking others, reading manuals, participating in courses)*

⋮

How have you experienced the transition from the old to the new system?

Figure 1: *Example from questionnaire used in the research project.*

We decided to pose questions about four main subjects: (1) The system development process; (2) the present experienced reality with respect to use of the standard system; (3) visions about future changes to the standard system; (4) general demographic questions about the interviewee. The structure of the questionnaires directly reflected activities mentioned in project plans and notes from project coordination meetings. Questions related to the system development process were for example subdivided into four categories: (1) Test and selection of standard system; (2) design of new or changed functionality; (3) design of user interface and (4) design of work procedures.

The acquisition of situational knowledge helped us in three important ways. Firstly, it made it possible for us to pose questions in terms familiar to the interviewees. We knew for example that the system developer used terms like "data model" and "testing model". Secondly, we were able to design more efficient interviews than would otherwise have been possible. The last question in figure 1 illustrates this. It presumes the transition from an old to a new system and focuses on properties of this transition. Without any situational knowledge we could not have prepared such a question in advance and we would have had to spend valuable interview time to be aware of the transition. Thirdly, we were able to design questionnaires specialized to the actual work situations of the participants. We ended up with two questionnaires - one for the system developer and one for the three user representatives. Our situational knowledge indicated that the same questionnaire could be used in all user interviews.

We enhanced the quality of the questionnaires by means of reviews and inspections.

Letting fellow colleagues evaluate questionnaires turned out to provoke a valuable debate about relevance of questions and their formulations. As a part of the improvement process we tried to imagine how respondents would feel when confronted with the questionnaires. Among the visible results of this was an improved sequencing of the questions. It took about one week's work for us to design and review the questionnaires.

3.4 Doing interviews

We performed the interviews in a room near to but separated from the interviewees' daily working environment. Our recording equipment (tape-recorder and microphone) were the most important part of the physical environment in the interview situations. Three persons were present during each interview: An interviewee, one of us doing the interview and the other taking notes. The note-taker generated and captured interpretations and asked supplementary questions. We interchanged our roles as interviewer and note-taker half way through the interview with the system developer because of the length of this interview.[6]

In order to make the interviewees relax we initiated each of the four interviews by an introduction of ourselves, the research project and the purpose with the interview. We emphasized the importance of their participation and we explained that the interview transcripts would be confidential and that names and places would be anonymous when the data was used and interpreted. By doing this we enabled the interviewees to decide whether they would allow us to tape-record the interviews or not. None of the participants refused the interviews to be taped.

According to the questionnaires we intended to structure the interviews as an iteration of the following pattern: (1) We read a question to the interviewee strictly as written in the questionnaire; (2) the interviewee formulated an answer in his or her own terms; (3) if necessary we formulated supplementary questions based on the probes. For a number of reasons the reality in the interview situations deviated from this scheme.

We were not very successful in our attempts to read questions to the interviewees properly. We often posed a question twice in different wordings before the interviewee had a chance to answer. The reason is probably that we were not sufficiently aware of the time needed for the interviewees to formulate answers. As a consequence they did not always get the time necessary to think about the questions, it was not always clear for them what the questions were actually about and consequently later on the process of transcribing from tape to word processor turned out to be unnecessarily time consuming.

Missing links between questions and answers was not an unusual situation. Sometimes the interviewees used a question as an opportunity to say whatever they had in mind. It happened especially often in the interview with the system developer. He had been the only system developer in the project and needed to talk about his frustrations. Actually this gave us valuable information about the system development process that we had not

[6]The interview with the system developer took approximately 3 hours, and each interview with the users took about 1 hour.

foreseen when we designed the questionnaires. Our challenge was to handle such situations without loosing the control over the interviews and to be sure that the stated question actually was answered. Sometimes the interviewees did not understand a question and we had to reformulate it in order to clarify. During the interviews we decided to skip some questions because the interviewees had already covered the subject. Figure 2 contains a small part of an interview.

Interviewer:
Which subproducts were produced in these activities, for example descriptions or other things related to design of functionality?

Interviewee:
The documentation is not too good. The part made by the consultants is only partly documented. The part made by the supplier is well documented.

Interviewer:
Also the changes they made?

Interviewee:
Yes, but most of it is technical descriptions.

Figure 2: *Example from interview with system developer.*

To summarize we had to plan during the interviews. How should we make the meaning of a question more clear? To what extent should we let the interviewees avoid to answer a specific question? How could we control the interview and at the same time maintain a feeling of relaxation? The probes written after each question in the questionnaire turned out very useful in situations where we had to reformulate questions because the interviewees did not understand the original ones or to create new questions if the answers did not fulfill our expectations. Our situational knowledge helped us understand the answers, improvise supplementary questions and control the interviews situations in a smooth way.

One last point regarding the three user interviews is worth mentioning. We wanted the users to tell us about their experiences with the system development process but directly asked about this they just stated that they did not participate. When we asked them about use situations their answers showed that they in fact had participated in the system development process. They just were not aware of this and thus could not tell about it when directly asked. The questionnaires expressed our preunderstanding of system development and use which differed from the users understanding. In this situation it was important for us to be able to continue the interviews on the users premises.

3.5 Documenting data

We realized from the beginning of the research project that the whole idea of the qualitative approach could get lost if we did not document the data carefully. We planned our documentation work by answering the following questions: Which activities should be documented? Why and how should these activities be documented? As an integrated part of all activities we took notes in order to record ideas and impressions. Besides serving as extended memory the note-taking provoked more conscious reflections in the situations. In order to emphasize the two important documentation activities we describe how we documented the situational knowledge and the interviews.

Documenting situational knowledge: We designed note-cards in order to record important data found when reading documentation from the system development process. The note-card shown in figure 3 is reproduced in order to give an impression of the structure and possible content of such cards. This specific note relates to the user manual for the standard system. We ended up with about 100 note-cards and they turned out to be useful in the later work. For example we used them when we prepared the interviews and they were a help to us when describing the system development process.

Title:	Communication/decision.
Time:	26/2 1987.
Place:	T-Purchase.
Source:	Supplement 4, report from meeting 18/2 1987.
Actors/roles:	From system developer to project manager.
Note:	**Status report, Purchase Department:** "It is becoming more and more demanding to getting activities in relation to interfaces between the different subsystems started. A report from *Victor* is therefore a necessity. This report is wanted! "

Figure 3: *Example of a note-card.*

Documenting interviews: We tape-recorded all interviews in order to enable later use of direct quotations that precisely expressed what was said during the interviews. While listening to the tapes we used a word processor to transcribe the resulting 7 hours of recorded talk into about 265 typed pages (Bækgaard & Sørensen 1988). Afterwards we checked the transcribed material against the contents of the tapes, again using a word processor. As an experiment we used pen and paper as a temporary medium between tapes and word processor because of our poor typing skills, but we did not get anything in return for the extra work effort. Transcription of tapes to word processor and checking the results took about two weeks full-time work for us.

We wanted the documented interview data to be a relatively precise expression of what was said during interviews and we wanted as far as possible to defer the process of interpretation to stages of using and interpreting the qualitative data. We therefore transcribed all tape recordings in full length letting the typed documentation of the interviews contain every spoken and unfinished sentence. We had to leave out some details from the interviews - for example tonal variations in speech and thinking pauses cannot be seen in the written documentation. Figure 2 contains an example of an interview transcription (Bækgaard & Sørensen 1988, p. 68).

We used notes taken during the interviews to make summaries which was to be used as guidelines in the work of using and interpreting the qualitative data. The summaries, however, turned out to be of little use because of the relative small time span between interviews and transcription. We did not need the summaries as extended memory because we made the transcriptions before we lost the impressions from the interview situations.

3.6 Using and interpreting data

In this section we give examples of different ways we have used the qualitative data from our research project i.e. as direct quotations, as reformulated quotations and as a basis for generalization. We used the knowledge about the system development process gained during interviews, informal talks, system demonstration and documentation reading in the process of selecting and interpreting the qualitative data.

As an example of the use of reformulated quotations we have used interview transcriptions and the note-cards to characterize the process of choosing, modifying and adapting a standard system and bringing it to use. During one of the interviews the system developer said:

> One couldn't say that I in fact made tests of other products - that is to say - I looked at other products but we didn't bring them in the house to test them in a test installation (Bækgaard & Sørensen 1988, p. 44).

This statement was reformulated and used as part of a description of the process of choosing a standard system:

> He did not test other systems than the *supplier*'s. He looked at other systems but did not put them in a test installation in the purchase department (Sørensen 1989a, p. 49).

This description is meant to be a interpretation of the statement in the sense that the idea is to emphasize the interviewees point of view in a readable manner.

Another example show the use of interview statements as a basis for generalization. The following is also a statement made by the system developer:

> What the users have in there is small notes which they mostly have made by themselves describing exactly the functions they need. When they come to me and ask about something they always bring paper and pen. Then they write down on their own paper whatever I say. Nobody ever reads the big Bible, never (Bækgaard & Sørensen 1988, p. 69).

The system developer explained how the users actually wrote their own manuals instead of reading the prewritten ones (referred to as "the big Bible"). One of the ideas developed from the system developers description of how the users learned to use the system was that this ought to be seen as the normal way to produce usable manuals (Bækgaard 1988). Instead of complaining about the little use of prewritten manuals users should be encouraged to write their own manuals in a systematic way. Besides the greater relevance of such manuals compared to prewritten ones the mere writing could help the users to improve their reflection in problem situations.

These two examples illustrates different ways of using qualitative data. In the first example the data are used as a basis for a description of a system development process and the role of the quotation is first of all to insure that the description is reliable. In this example the point of interest is the situations mentioned in the interview. In the second example the quotation is used as a source of inspiration. The point of interest is the creation of ideas to be used in other situations than the one mentioned in the interview. In general transcribed material was used in a combination of reformulation and generalization. Further more direct quotations was used for illustrative purposes without reformulating or generalizing.

4 Conclusion

One of the basic assumptions behind our discussions has been that empirical approaches necessarily have to play a major role in system development research. Based on our own experiences we have argued for the use of qualitative interviewing as a valuable approach in this context and we have identified six interrelated activities spanning the most important aspects of qualitative interviewing: (1) Preliminary research design; (2) Acquiring situational knowledge; (3) Designing questionnaires; (4) Doing interviews; (5) Documenting data; (6) Using and interpreting data. Besides we have argued for the importance of situational knowledge as an important mean to improve the quality of the research activities. The content of the six activities can be summarized in the following way:

Preliminary research design: The search for candidate organizations ought to be initiated as early as possible because it can take a long time to make final arrangements. Besides this we recommend that an overall plan of the research project is made including decisions about eventual use of supplementary research approaches.

Acquiring situational knowledge: The situational knowledge shall make it possible for the researcher to design efficient questionnaires and to understand the interviewees statements. Situational knowledge can be acquired by means of informal talks, document reading and observation.

Designing questionnaires: The questionnaire shall enable the researcher to control the interview and at the same time allow the interviewee to say what he finds important expressed in his own terms.

Doing interviews: The purpose is to produce the qualitative data in an interview based on the questionnaire. The researchers ought to follow the questionnaire closely in order to control the interview situation but at the same time be prepared to let the interviewee bring up interesting subjects. It is very important to pose questions clearly and to give the interviewee sufficient time to answer.

Documenting data: The purpose is to capture the qualitative data and enable later use of these. If possible the researcher should tape-record all interviews and transcribe the important parts to paper. Note-taking can be very useful when the researcher acquires situational knowledge and it can help him to decide which parts of the tapes is worth transcribing.

Using and interpreting data: All other activities is done in order to be able to use the data for research purposes. Interview data can be used in at least three different ways - direct quotations, reformulations of quotations and generalizations of quotations.

Our main advice to other researchers who have decided to use a qualitative interviewing based approach to system development research can be expressed as follows: (1) Plan the research process using the six activities as a guideline; (2) be sure to acquire and use situational knowledge in order to improve the quality of the research; (3) get detailed ideas from sources like the preceding section 3, (Patton 1980) etc; (4) decide how to implement the research process in terms of time, space, people and other resources.

The concept of empirical research is more fundamental than explicit emphasized by our discussions. Firstly, our approach to empirical research theory has as a matter of fact been empirical. We have used our experiences in qualitative interviewing as an empirical background for discussions of aspects of the approach. This is by no means a coincidence - empirical research theory should to the same extent as system development theory be based on empirical research.

Secondly, research theory should play two roles in relation to system development. Besides the role discussed in this paper research theory should be seen as an important part of system development theory. Among other activities system development deals with inquiries into complex social worlds. Therefore research theory should help systems developers improve their skills related to such inquiries.

References

Andersen, N.E., F. Kensing, M. Lassen, J. Lundin, L. Mathiassen, A. Munk-Madsen and P. Sørgaard: *Professionel systemudvikling (Professional Systems Development).* Teknisk Forlag. København, 1986. In Danish.

Andersen, P.B. and K.H. Madsen: "Design and Professional Languages". (From K.H. Madsen (ed.): *Sprogbrug og design (Design and Language Usage).* Ph.D. Thesis, Daimi, Århus University, 1988.). Århus, 1988.

Borum, F. and H. Enderud: *Konflikter i organisationer - belyst ved studier af edb-systemarbejde (Conflicts in Organizations).* Nyt Nordisk Forlag Arnold Busck. København, 1981. In Danish.

Bryman, A.: *Quantity and Quality in Social Research.* Unwin Hyman. London, 1988.

Bækgaard, L.: *Edb-systemers kvalitet - empirisk og teoretisk baserede reflektioner (The Quality of Computer Systems).* Master of Science thesis in Computer Science, Aalborg University. Aalborg, 1988. In Danish.

Bækgaard, L.: "Understanding Information Systems Quality". Forthcoming. Aalborg, 1989b.

Bækgaard, L.: "Information Systems Work - Looking Beyond Development Projects". Forthcoming. Aalborg, 1989c.

Bækgaard, L. and C. Sørensen: *VRID. Vision og realitet i design - to systemudviklingsprojekter belyst ved interview af brugere og systemudviklere (Vision and Reality In Design).* Internal Project Documentation, Aalborg University. Aalborg, 1988. In Danish.

Ehn, P. and M. Kyng: "A Tool Perspective on Design of Interactive Computer Support for Skilled Workers". Århus University, Computer Science Department, DAIMI PB - 190. Århus, 1985.

Etzerodt, P. and K.H. Madsen: "Methodological Choices when Evolving Users' Understanding of the Impact of Computers.". (From Lassen, M. and L. Mathiassen (ed.): *Report of The Eighth Scandinavian Research Seminar on Systemeering, Aarhus, August 14.-16. 1985.* Aarhus University. 1985.). 1985.

Feldman, M.S. and J.G. March: "Information in Organizations as Signal and Symbol". (From R. Galliers (ed.): *Information Analysis. Selected Readings.* Addison-Wesley. 1987.). 1981.

Holmquist, B.: "Work, Language and Perspective. An Empirical Investigation of the Interpretation of a Computer-Based Information System". To appear in Scandinavian Journal of Information Systems vol. 1. 1989.

Jepsen, L.O., L. Mathiassen and P.A. Nielsen: "Back to Thinking Mode - Diaries as a Medium for Effective Management of Information Systems Development Projects". Århus University, Computer Science Department. Århus, 1986.

Kasanen, E. and R. Suomi: "The Case Method in Information Systems Research". The Finnish Journal of Business Economics, special edition 4-1987. Helsinki, Finland.

Mumford, E., R. Hirscheim, G. Fitzgerald and T. Wood-Harper (ed.): *Research Methods in Information Systems.* North-Holland. Amsterdam, 1984.

Munk-Madsen, A.: "Viden om systemudvikling (Knowledge on Information Systems Development)". Århus University, Computer Science Department, DAIMI PB - 213. Århus, 1986. In Danish.

Patton, M.Q.: *Qualitative Evaluation Methods.* Sage Publications. U.S.A, 1980.

Sørensen, C.: *Standardsystemer i organisationer - valg, modificering, indpasning og ibrugtagning (Standard Systems In Organizations).* Master of Science Thesis in Computer Science, Aalborg University. Aalborg, 1989a. In Danish.

Sørensen, C.: "Standard Information Systems In Organizations – Rational and Political Interpretations of a Case". Forthcoming. Aalborg, 1989b.

Wynn, E.H.: *Office Conversation as an Information Medium.* Department of Anthropology, University of California. Berkeley, 1979.

TWO CONCEPTIONS OF DESIGNING

Kathy Carter

Rank Xerox Cambridge EuroPARC

61 Regent Street

Cambridge CB2 1AB

England

Abstract: In this paper I explore two ways of designing, caricatured as those of the "handy-worker" and the "architect". The contrast is between fixing up quick, local solutions for problems as they arise and making large-scale plans that determine everything in advance. Traditionally software designing has tended to follow the architect, often resulting in systems that fail to meet the needs of the users. In response to this there is a trend towards involving the users in the design of systems. This opens up an important role for the handy-worker's style of designing. I illustrate the interaction between these styles of working from a software development project undertaken at EuroPARC.

Introduction

In designing it is possible to distinguish the approach of the "handy-worker" from that of the "architect". The handy-worker works piece-meal, fixing up quick solutions to specific problems as they arise whereas the architect makes large-scale plans that lay out in advance what will be done.

I am going to argue that the approach of the handy-worker has an important contribution to make to software designing. The handy-worker responds quickly and locally to whatever he finds, not relying much on reflection or planning. By being closely involved with the users he is in an ideal position to adapt and evolve systems to fit the users better. Traditional systems design tends towards an architectural approach of analysis and specification followed by implementation and finally use of the resulting artifact. Unfortunately this approach often results in systems that are not very suitable for their users. A way of tackling this problem is to involve users more actively in the design of their technology. Following the handy-worker rather than the architect we find a way of designing that is well suited to encouraging user involvement. I suggest that the handy-worker and architect have complementary contributions to make to the design process, and that we need an approach to designing that makes space for both.

Before I elaborate on these ideas I would like to recount an experience drawn from the "Buttons" project at EuroPARC that highlights these differences for me. Buttons are a mechanism for encapsulating complex functionality in simple screen objects. They can be created to support many different tasks for the users. The aim was to provide a way for users to tailor their workstations to suit their particular tasks by creating and modifying Buttons for themselves. The Buttons software was given to everyone within EuroPARC, including non-technical administrative staff. This story centres around Mary, one of the administrative staff who got involved in the Buttons project. The other players are Ann and Jane, who were working full time as designers on the project.

The story of the TEdit Button

When Mary wanted to edit a file, she would search for the file in a "filebrowser" that gave her a list of file names in a particular category. She would then call the edit program on that file. The program created a window on the screen where she could work on the file. If she needed more space on screen temporarily then she could select "shrink" from the window's menu. This made an "icon", a small object on the screen that replaced the full size window. When she wanted to start working on the file again she could expand the icon to get back her editing window. There was no need to find the file and start the editor again. Unfortunately, if she closed the editing window down completely, she then had to search for the file again or remember its full name to edit it again. (see fig. 1)

Then she was given some "Buttons", which were small rectangular objects on her screen that had commands and information associated with them. Mary's Buttons provided short-cuts for editing files. If she hit one of these Buttons it would call the editor for her on the file associated with that Button. Unlike the icon the Button stayed on the screen all the time, whether or not she was currently working on the file. The file did not have to remain open to keep the Button alive. As long as the Button was there she had no need to go back to search in a filebrowser for the file. (see fig. 2)

Mary's Buttons had been built specially for files that she used regularly. In additio, the Buttons designers had designed her Buttons to be easily copied and modified to edit other files. She could hold down the "Copy" key and click on the Button to make a copy. By holding down the "Edit" key and clicking she could get a menu of items she could change (see fig. 3). She could select the filename parameter and so set up the new Button to edit another file. If she was feeling more adventurous she could also change the action of the Button so that, for example, it printed the file rather than editing it.

Jane, one of the designers, then met with Mary to talk about how she worked with files and to see whether Buttons could be made more useful for her. It turned out that she never tried to copy or modify her Buttons, even though Buttons for other files would have been very useful to her. She edited other files in the same way as before, using the filebrowser to locate them. It quickly became apparent to Jane that the idea of copying and modifying Buttons didn't make any sense to Mary. Together they explored the way Mary tackled various tasks and played through other ways of creating Buttons. They came up with the idea of creating Buttons directly from the editing window. While Mary was editing the file in the usual way could she somehow say "make a button for this"? There would be no need to break off from what she was doing to move into the world of copying and modifying Buttons.

As a result of this discussion, Ann (the Buttons implementer) changed the "shrink" option for the editor window so that it created a Button instead of an icon. Jane and Ann, as designers, could see the potential for confusion between the behaviour of icons and Buttons. Also, this development did not follow on from any particular part of their project plans, but was a "one-off" response to the users. But Mary and her colleagues were very pleased with this facility and rapidly incorporated it into their work. They are now using it very heavily and are not bothered by the possible confusions.

Ann recently produced a new implementation of the underlying Buttons program and this provided an opportunity to re-visit the design of the editor Button. Jane wanted to retain the functionality as it had proved so useful to Mary, but Ann wanted to integrate it better into the rest of the Buttons world. They decided to separate "Shrink" and "Make Button" so as to remove the icon / Button inconsistency. By changing "Shrink", which was a basic system command, they had entangled Buttons functionality with the underlying system. A separate "Buttonize" command means that Buttons now "sit on top of" the rest of the system.

It then became apparent to Ann that this "Make Button" mechanism made sense for windows other than editors. For example, the command used to create a file-browser window could be captured in a Button. This Button could then be hit to re-create that browser. All of the window menus now have "Buttonize" as a command that can be applied to the window. A button is created that will re-create that window if it is possible for that type of window.

Behind the story

As the Buttons project progressed the designers were confronted by a problem. They could see that Buttons were very little used in the ways they had expected. The example of Mary and her editing Buttons is just one example of someone not making use of the functionality of the initial implementation. It looked to the designers as if Buttons could help her in her work, but she was not taking advantage of them. How could they redesign Buttons so that they were more useful to people like Mary?

One way to develop more suitable technology is to involve the users themselves in the design process. This approach has been particularly pioneered in Scandinavia [2] as a way of improving democracy in the workplace. An important aspect of this work is enabling people without training in software design to understand the implications of particular technical decisions. For example, in the Utopia project paper and cardboard models were used to help graphic workers experience what it would be like to use a computer-based tool for page makeup. Rapid prototyping that allows people to have early experience of any new ideas is also important. The Buttons team decided to take this approach of user-involvement. This in turn changed the roles of people within the team and influenced the shape of the design process.

As Jane began to work with Mary and the other users, distinctive roles began to emerge within the design team. Jane found herself working as a handy-worker, producing bits and pieces that met particular immediate needs that people had. Sometimes these were very small and easily done. Others, such as creating a Button directly from the editing window, were more difficult and needed the involvement of Ann, the Buttons architect and implementer. From the point of view of Ann's plan for the structure of Buttons this idea for creating Buttons was out on a limb. But it fit well with the way Mary and her colleagues organised their work and was easier for them to understand than making a Button "in the abstract" for future use. As a result, Ann implemented the facility and it proved a great success with the users.

Once Jane and Ann saw how successful this facility was, they attempted to integrate it better into the rest of the system. Ann was concerned that the structure of Buttons and their relationship to the underlying system be clean and consistent. She developed a general "Buttonize" mechanism that gives users a way to create Buttons for a variety of different windows, extending their ability to create Buttons without the need for programming. This new development arose from the interplay of Jane's work with the users and Ann's concern for maintaining a consistent architecture.

Two Conceptions of Designing

Ann's role as a system architect is one that we can easily recognize as part of the software design process. Jane's role as a handy-worker, on the other hand, tends to be overlooked. To see why, we must look at different ways in which we can characterise the design process.

"Design" often has connotations of coordination and perfection. The solution is planned and specified so that the parts fit together to make a coherent whole, fitting some overall scheme. We can see this in the design and construction of buildings. For example, there are building projects where the requirements are analysed in depth and detailed plans and specifications are produced by the architects. The proposed solution can then be analysed to check that it meets the various criteria that are required for such a building. The building is then constructed according to the plan.

But does "designing" have to be equated with"planning" and "specification"? It could as easily be the local handy-worker fixing up our kitchen with whatever comes to hand. He probably does not have much in terms of a "grand plan" but rather what he does grows and evolves in response to the situation. If it's not quite right then he will make changes or maybe start again. We would probably not ask the handy-worker to build us a new house but we may well ask him to add some new light fittings or install a shower in our architect-designed house. What we see here is two different conceptions of designing buildings - the approach of the "architect" as opposed to that of the "handy-worker".

The software architect

If we try to look for these two approaches in software designing then we can see that the traditional ideal for this process has been what I have called the architect's approach. Project structures reflect a conception of designing as a process of precise specification followed by implementation. There are various models of the stages that a project must go through and of the mile-stones that can be used to assess progress. Typically a feasibility study is followed by systems analysis and then requirements specification. Once the requirements are specified the system is designed, programmed and then tested before installation. Each stage in this process has particular products that must be completed before moving on to the next stage (for example [3]).

A professional designer will have various analysis techniques that he can bring to bear on the users' situation in order to characterise the requirements of the system. Detached and objective consideration of competing requirements is striven for. Ideally he will find a theory that captures the salient features of the situation and enables him to make the correct decision as to what is best for the user. Eason [1] outlines the characteristics of a variety of design methodologies. Some do not explicitly consider the user at all, but concentrate on such things as the flow of information. Other methodologies might bring the user into the design process, but as a passive object for observation and analysis. In structured design methodologies the users are involved by being given the chance to react to the design as it passes through the various stages. Even so the aim is still to produce a specification for the system to be implemented and the system is essentially fixed once it is built.

The traditional way of designing takes a very static view of the users' situation, and ignores the active process of "making use" of a new technology. The technology can be designed to fit, to be the best possible given the analysis techniques. But once the technology is designed and installed the users have to incorporate it into their work practices. The workers must build up a relationship with the new technology, something that does not happen instantaneously. This very process changes the setting and may invalidate the original analysis.

The results of this can be seen in the often lengthy process of "maintenance" that follows the installation of a new system. During this period the system is adjusted to suit the way the work is really done. The actual users can throw up all sorts of problems with the beautifully planned system. Eason [1] lists problems ranging from failure to support adequately the users' tasks and unwelcome changes in job content through to effects on overall organisational policy. The architect's response to this is to look for better methods of analysis and planning so that, next time, problems can be anticipated and dealt with. But this detached and abstract approach will always fail if it ignores the fact that the users are actively involved in their work situation.

The system handy-worker

The system handy-worker does not officially exist in traditional software engineering. By analogy with our building example he is someone who is locally available and can fix things up and provide solutions whenever people need them. He is probably part of the local community and so speaks its language and understands its priorities. Because of this he is in the position to produce solutions that are well suited to the people using them. He probably would not build large new systems but would build small bits and pieces to help people with particular tasks. By designing and building "in place" along with the users, he gives them the opportunity to be involved with whatever is being built right from the start.

Involvement rather than reflection is more characteristic of the handy-worker's role. He is involved in the users' situation and is in turn in the ideal position to involve them actively in the design process. He can respond to their requests and experiment with them to find new ways of working. Because he is not bound to producing specifications or long-term plans he can be flexible in what he does. The users can respond to what he produces at a very early stage and so are in a better position to keep the design in line with what they need.

On this small scale the handy-worker is very successful but the lack of planning can bring a variety of problems on a larger scale. A shanty-town, constructed piece-meal with no plans, is an immediate response to peoples' needs but the lack of services that could have been provided by central planning is a serious difficulty. In software design, systems like Unix™ and Interlisp are substantial systems that were "grown" by the users in a similar unplanned way [4]. For those within the community the systems are wonderful and ideally suited to those who developed them. For anyone outside this community they are confusing, inconsistent and difficult to learn.

Bringing the two approaches together

The handy-worker is able to respond directly to the users' needs and to fit things to the work setting as he finds it. But this can bring the danger of increasing chaos with large numbers of uncoordinated changes. On the other hand, static, architect-designed software runs the danger of being useless and inflexible. The structure may be beautifully clean but if it cannot be moulded to people's real working situation then it is no good.

In reality, we often do see both approaches at different times in a project. For example, the handy-worker's style of work may be seen in the post-hoc fixes that make a system usable. The problem is that we see this as an embarassment rather than a positive feature of the design process. If we are serious about involving users throughout the design process then the role of the handy-worker assumes even more importance. By explicitly recognising this role we can then draw together the complementary contributions of the two approaches.

Concluding remarks

I started with the story of the "edit button" that illustrated the creativity and also the tensions arising from involving users actively in designing their technology.

The caricatures of the "handy-worker" and the "architect" seemed best to capture the distinction between the different approaches to designing going on in this project. Jane's role as "handy-worker" developed in response to our desire to involve users in the design of Buttons. Ann retained the more conventional "architect's" role within the team. Although these were the predominant roles that they took, the roles were more linked to the context than to people. Jane worked as a handy-worker with the users but then shifted to be more of an architect in the design team meetings. Ann took on some of the handy-worker's approach when she worked with Jane to produce solutions to the users' needs.

The progress of the Buttons project cannot be described by a linear process of specification followed by implementation and then use. A more appropriate way of seeing it is as parallel strands of specification, implementation and use. Each strand is intertwined with the others and has an influence on the overall process.

The priority for the architect, as I have characterized her, is for a clear plan and for a clean structure. On the other hand, the handy-worker made the users' needs a priority. These different priorities often suggested different directions for the work to be done. The answer was not in a compromise between the two, but with acknowledging both extremes and working with the tensions and conflicts that this produced. We found solutions that transcended the differences and resulted in new and creative design ideas.

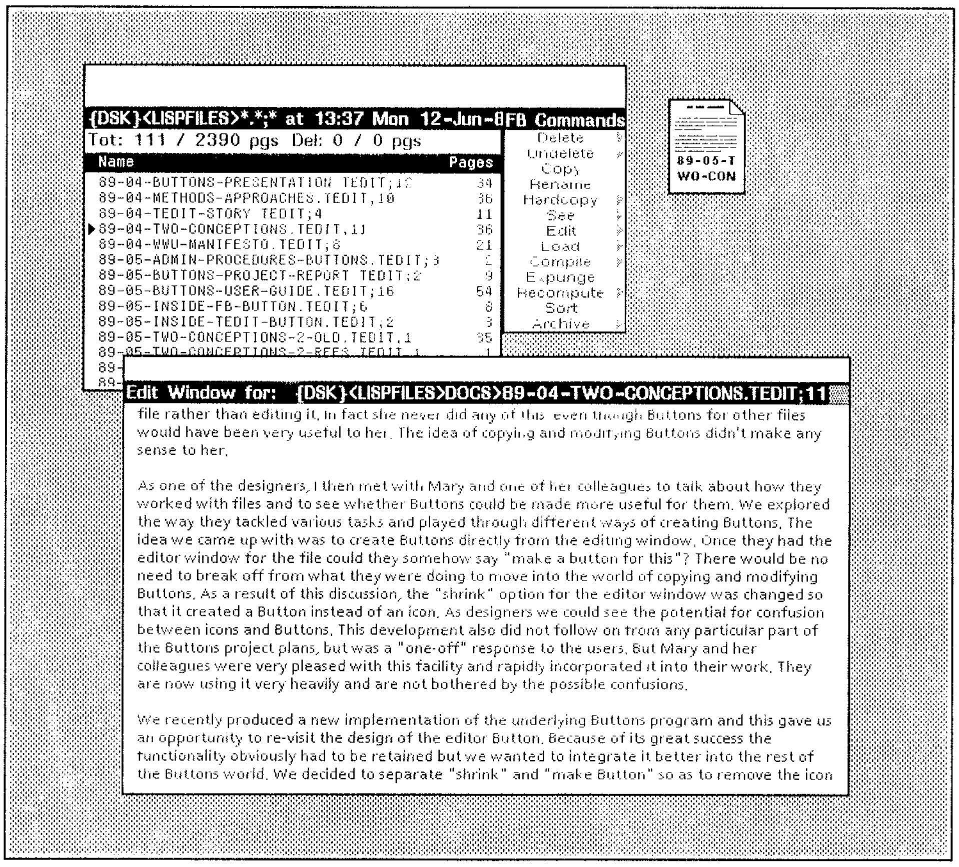
References

[1] K. Eason, "Information Technology and Organisational Change", Taylor and Francis, 1988

[2] C. Floyd et al, "Out of Scandinavia", Technical University of Berlin, Dec. 1988

[3] P.W. Metzger, "Managing a Programming Project", Prentice-Hall, NJ., 1981 (Second Edition)

[4] W. Teitelman and L.Masinter, "The Interlisp Programming Environment", Computer 14:4, April 1981


```
{DSK}<LISPFILES>*.*;* at 13:37 Mon 12-Jun-8
Tot: 111 / 2390 pgs  Del: 0 / 0 pgs
Name                                          Pages
 89-04-BUTTONS-PRESENTATION TEDIT;12             34
 89-04-METHODS-APPROACHES.TEDIT,10               36
 89-04-TEDIT-STORY TEDIT;4                       11
▶89-04-TWO-CONCEPTIONS.TEDIT,11                  36
 89-04-WWU-MANIFESTO.TEDIT;8                     21
 89-05-ADMIN-PROCEDURES-BUTTONS.TEDIT;3           2
 89-05-BUTTONS-PROJECT-REPORT TEDIT;2             9
 89-05-BUTTONS-USER-GUIDE.TEDIT;16               54
 89-05-INSIDE-FB-BUTTON.TEDIT;6                   8
 89-05-INSIDE-TEDIT-BUTTON.TEDIT;2                3
 89-05-TWO-CONCEPTIONS-2-OLD.TEDIT,1             35
 89-05-TWO-CONCEPTIONS-2-REFS TEDIT,1             1
 89-
 89-
```

Edit Window for: {DSK}<LISPFILES>DOCS>89-04-TWO-CONCEPTIONS.TEDIT;11

file rather than editing it, in fact she never did any of this even though Buttons for other files would have been very useful to her. The idea of copying and modifying Buttons didn't make any sense to her.

As one of the designers, I then met with Mary and one of her colleagues to talk about how they worked with files and to see whether Buttons could be made more useful for them. We explored the way they tackled various tasks and played through different ways of creating Buttons. The idea we came up with was to create Buttons directly from the editing window. Once they had the editor window for the file could they somehow say "make a button for this"? There would be no need to break off from what they were doing to move into the world of copying and modifying Buttons. As a result of this discussion, the "shrink" option for the editor window was changed so that it created a Button instead of an icon. As designers we could see the potential for confusion between icons and Buttons. This development also did not follow on from any particular part of the Buttons project plans, but was a "one-off" response to the users. But Mary and her colleagues were very pleased with this facility and rapidly incorporated it into their work. They are now using it very heavily and are not bothered by the possible confusions.

We recently produced a new implementation of the underlying Buttons program and this gave us an opportunity to re-visit the design of the editor Button. Because of its great success the functionality obviously had to be retained but we wanted to integrate it better into the rest of the Buttons world. We decided to separate "shrink" and "make Button" so as to remove the icon

Figure 1: A Filebrowser, an Edit Window and an Icon

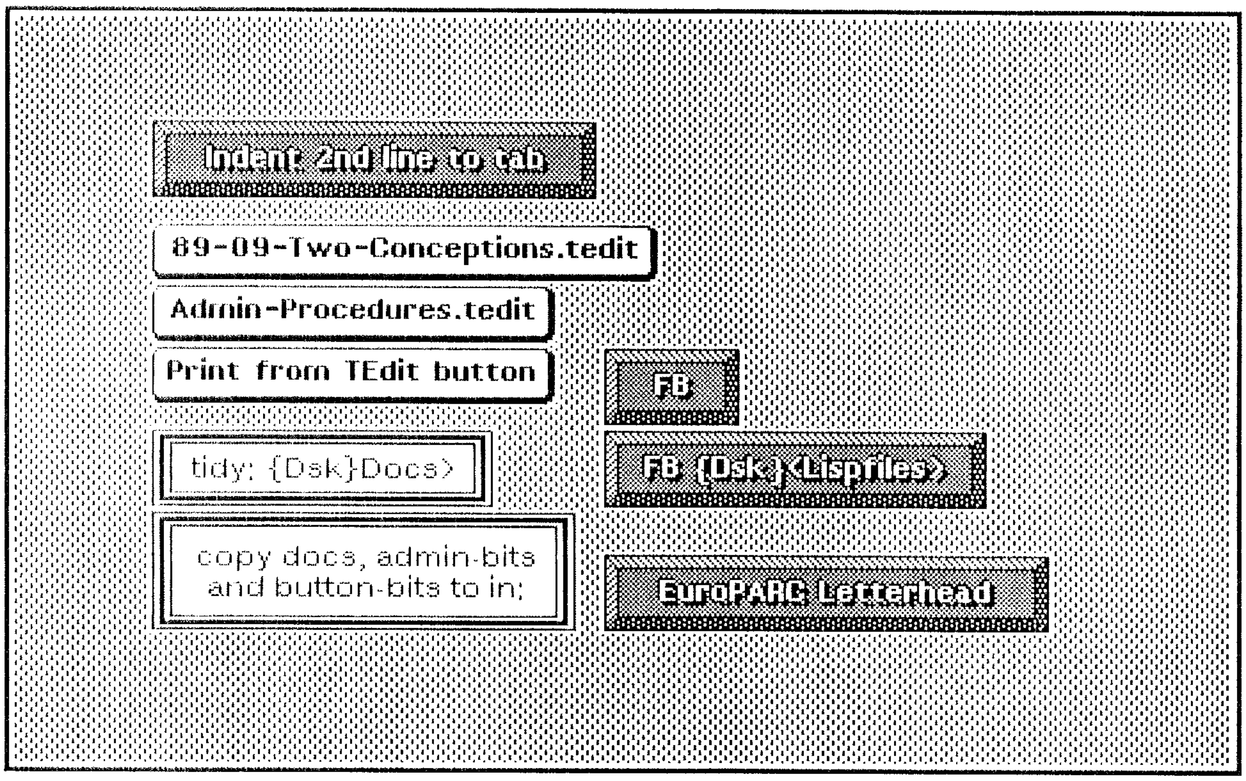
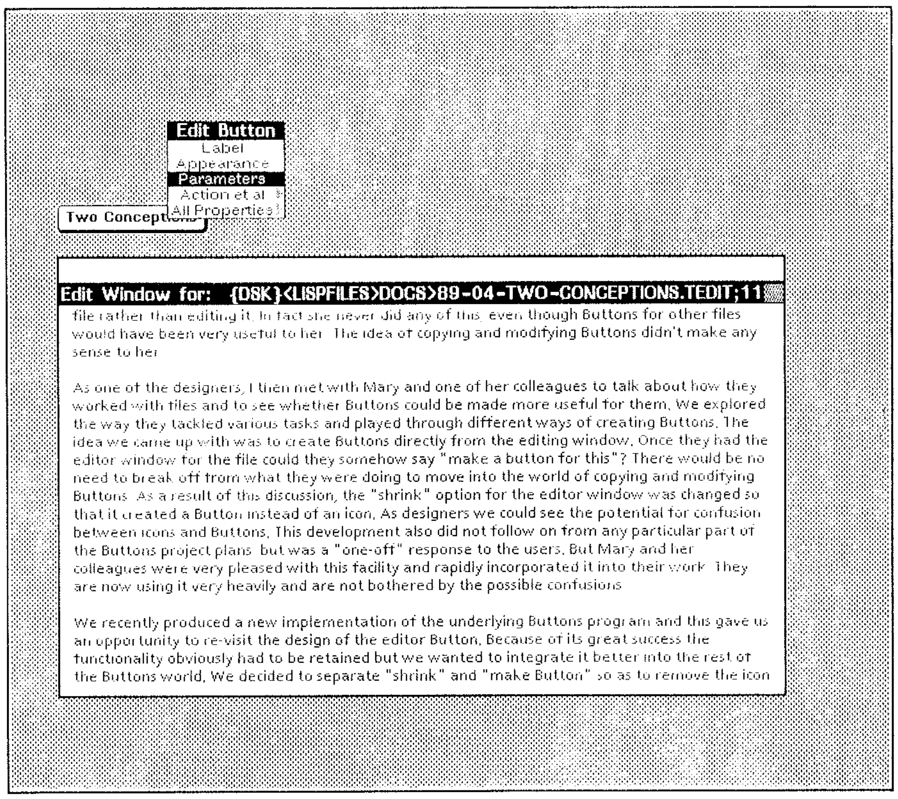


Figure 2: Various buttons, including some for files (89-09-Two-Conceptions, Admin-Procedures, EuroPARC Letterhead)



Edit Window for: {DSK}<LISPFILES>DOCS>89-04-TWO-CONCEPTIONS.TEDIT;11

file rather than editing it. In fact she never did any of this, even though Buttons for other files would have been very useful to her. The idea of copying and modifying Buttons didn't make any sense to her.

As one of the designers, I then met with Mary and one of her colleagues to talk about how they worked with files and to see whether Buttons could be made more useful for them. We explored the way they tackled various tasks and played through different ways of creating Buttons. The idea we came up with was to create Buttons directly from the editing window. Once they had the editor window for the file could they somehow say "make a button for this"? There would be no need to break off from what they were doing to move into the world of copying and modifying Buttons. As a result of this discussion, the "shrink" option for the editor window was changed so that it created a Button instead of an icon. As designers we could see the potential for confusion between icons and Buttons. This development also did not follow on from any particular part of the Buttons project plans but was a "one-off" response to the users. But Mary and her colleagues were very pleased with this facility and rapidly incorporated it into their work. They are now using it very heavily and are not bothered by the possible confusions.

We recently produced a new implementation of the underlying Buttons program and this gave us an opportunity to re-visit the design of the editor Button. Because of its great success the functionality obviously had to be retained but we wanted to integrate it better into the rest of the Buttons world. We decided to separate "shrink" and "make Button" so as to remove the icon

Figure 3: Menu for editing a Button

AN ARTIFICIAL WORLD

An invitation to creative conversations on future use of computer technology

Bo Dahlbom
dahlbom@cs.chalmers.se
Department of Philosophy
Göteborg University

Lars-Erik Janlert
lej@seumdc51.bitnet
Inst. of Information Processing
University of Umeå

1. An invitation to creative conversation #1

Imagine that you were an electrician way back in the late 19th century, pondering the future use of electricity. What would you be able to come up with? Would you think of lighting and heating your house, cooking, lifting heavy things, as power to motors, welding, as information carrier? Or, would you be so stuck in the usage of your time that all you could think of was simple variations on use as a light source? How ill prepared for the future would you not be? And how could you possibly do a good job, as an electrician, in your own time, with such a shallow and mistaken conception of electricity?

You could always turn to philosophy, of course, to get a better understanding of where electric technology might take us: "Electricity is the purpose of the form from which it emancipates itself, it is the form that is just about to overcome its own indifference; for electricity is the immediate emergence, or the actuality just emerging, from the proximity of the form, and still determined by it—not yet the dissolution, however, of the form itself, but rather the more superficial condition, having not yet grown into independence of and through them." (Hegel, *Philosophy of Nature.*)

Now what do we do in system development? Aren't we stuck in our own time, or rather the time of the 60's, when we go on talking about computer technology use in terms of "information systems"? As if the computers still were the "data machines" of the 60's, in spite of the fact that our most spectacular research experiences in Scandinavia have been with computers as more or less UTOPIAn design tools. And design tools, whatever they are, certainly aren't information systems. The computer was used as a data machine in the 60's, and will continue to be so used in the future. But how important will that type of use be, compared to other types?

And what do the philosophers tell us? Are they much more to the point than Hegel was, as they go on warning us for losing our tacit knowledge, trying to teach us Heidegger and Wittgenstein?

In August 1989 the 12th Information systems Research seminar in Scandinavia was held in Skagen, Denmark. But it was only the second seminar under that name. What used to be called a "seminar on systemeering" changed its name two years ago. Was it not an unlucky change, even if IRIS is a nice acronym? To speak of current and future use of computer technology in terms of "information systems" seems to us as misleading as to speak of use of electricity in terms of "light bulbs." Why would one want to restrict the field of system design and development to dealing with information systems, in a time when such systems are crowded out by other uses of computer technology? The information society just came and went like other "societies" of that kind.

Technologies develop and so does their use. Technologies live their lives through cycles characterized by different kinds of attention. Examples of such cycles are: engineering, understanding, social consequences, environmental consequences, innovation. In an engineering cycle, resources are spent on developing the basics of the technology, on making it work. This cycle is often, as in the case of electricity, preceded by attempts to understand the basic phenomena underlying the technology. At other stages in the life of a technology it becomes more important to take seriously its (social and environmental) consequences, working back from them to change, restrain or even abandon the use of the technology. A third cycle is dominated by innovative efforts to diversify the use of a given technology. These cycles overlap, but it is still possible to distinguish them in the history of most technologies, in terms of what aspect of the technology that dominates attention in a certain community.

If we look at computer technology from the perspective of Scandinavian system development research, it is not difficult to see that the 60's was a period of engineering—the problem was to get the technology working. Börje Langefors worried about file handling and CPU-time. In the 70's, the consequences sailed up as the hottest topic. Kristen Nygaard agitated for democracy at work, and his disciples worried about deskilling. The 80's has moved from attempts at understanding to become a decade of use, of a growing interest in the explosion of applications and interfaces.

Being thus in the middle of a cycle of innovative use diversification, we would want to spend a few hours in a creative conversation about current and future use of computer technology. When we want to *understand* computer technology we ask questions like "What is a computer?," but when we're in a cycle of innovative use we'll ask instead "What can we make of computers?" And we are not looking for one answer, and not only for the general kind of answers: a tool, a medium for communication, an information system, etc. Such answers can at best provide starting-points for the more detailed answers we are interested in.

We live in an artificial world, and more and more of the artifacts, physical as well as social, of that world contain computer technology. What kinds of artifacts can we design in which computer technology plays interesting roles? That is, how can computer technology be used in producing a richer world of artifacts? That is the topic of our conversation #1, and it involves an active interest in the shaping of future computer technology. As system developers we are, perhaps, more used to dealing with a slightly different question, namely, how to put the already existing artifacts, containing computer technology, to good use. Here we want to have a go at being less passive, but we will return, more explicitly, to the latter question in the final sections.

We want to do with the computer what innumerable students have done in creativity tests with bricks: you are asked to enumerate as many possible uses for a computer that you can think of. But there is this difference: while the brick is a determinate object, the computer is much more of an open technical possibility. We are not asking only what we can *do* with the computer, but what we can *make* of the computer. The underlying assumption is the rather trivial observation that computer technology is, within its defining boundaries, what we (but who are we?) make it to be. And that we still are in the very beginning of computer technology use, with a wide open future ahead. And that we'd better hurry being creative, if we want it to be something really good.

In the vein suggested by Richard Rorty in the last chapter of *Philosophy and the Mirror of Nature*, we invite you to a conversation, not an inquiry, to abnormal rather than normal discourse, to a play with many perspectives rather than a search for objective truth. To get such a creative conversation going, we have tried to formulate some, rather general, categories for computer technology use. Whenever, in our conversation, our creativity falters, we can always go back to these categories and use them to help us in the generation of ideas. Or, when that fails, we can try our more analytical skills on criticizing, modifying and changing these categories so as to serve our creative purposes better. What principles could one use in such an attempt at categorization?

2. Categories of computer technology use

Technology use can be categorized in different dimensions, depending on the purpose with the categorization. Categories can be invented or they can be taken over from tradition. Categories can be developed from the top down, by looking closer at the general characteristics of technology use, or from the bottom up, by generalizing from actual examples of technology use.

Nurminen's (1988) categorization, systems-theoretical, sociotechnical, humanistic, is a categorization of use in terms of the power relation between the individual user and her computer. Nurminen describes it as a top down categorization, expressing the fact that the relation between man and computer can be varied in such a way that man is an appendix to the computer, or the computer to man, or they are partners in a (socio-technical) system. Nurminen needs his categories to distinguish three perspectives in system development, but it still is a categorization of computer technology use, and it gains its main interest from this fact. The different schools distinguished by Nurminen give different advice to system developers, based on their very different conceptions of how computer technology should be used.

There are only three categories in Nurminen's schema, and they are well worn. They are, in fact, the standard categories in the 19th century technology debate. In that debate, Romantic philosophers reacted to the inflexibility of the machine by suggesting more organic relations between man and his utensils, sometimes even turning back to pre-industrialized "crafty" use of technology.

Now, if one is really interested in taking a hard look at the possible use of computer technology, one cannot be satisfied with traditional categories like these. One can begin instead by asking oneself, top down, what it means to use technology. Use, was ist das? The first thing to notice is that the term "use" itself is misleading. Much of our interaction with technology (technical artifacts) is not well described as "use." But it is easy to be, like Heidegger, misled into thinking so, by examples like hammering, writing, etc. But think only of how we interact with cars: most of us use them as means of transportation, mechanics and gas station owners make a living out of them, clerks at the central automobile register keep files on them, driving instructors teach people to drive them, etc, etc. To say that the cab driver (the official) and her passenger (the citizen) both use the cab (social security system) tends to cover up rather than illuminate important "facts of use" (Janlert 1987). In spite of this, we will go on talking about "use," meaning "interaction" in general, sometimes using the latter term to avoid misunderstanding.

You use something as means to certain ends. That is why Kant finds it abhorrent to treat people as means rather than ends—that is to use them. It is easy, but maybe dangerous, to begin thinking of the use of artifacts as replacing or extending the use of bodily organs: shovels dig better than hands, with glasses we see better, a notebook is a memory substitute, etc. So, there we have two starting-points for a categorization: our ends and our natural means (organs). What are our ends? People like Maslow give us lists: eat, drink, shelter, sex, love, respect, fame, fortune, fun. These ends can themselves be categorized, (as in Zetterberg et al 1984) into ends of subsistence, consumption, and expression. Obviously these categories are too general to give us much help.

What are our natural means? We could (like they do in AI) categorize computer technology use in terms of what human capacities they replace or extend: movement, perception, memory, calculation, imagination, language, etc, but also with this kind of starting-point we are to far away from the bottom to be able to say anything worthwhile about computer technology use. Categories like these will not exactly send the mind into soaring flight. It leaves no empty boxes to fill in. So let us try to be a little more concrete.

An artifact in use plays a role, has a place in some human activity. We can categorize computer technology use by looking at different ways in which that technology relates to its environment. To use an artifact is to put it in play. The simple example is our causing a tool to have a certain effect. Artifacts can be categorized in terms of their causal side, in terms of their autonomy. Or they can be categorized in terms of their effective side, in terms of how they are geared to their environment. Some artifacts are geared to their environment physically, some symbolically.

We could develop a categorization based on how the computer fits into its environment, how it interacts with the world. We could classify its interaction domain in terms of the number and types of objects in it. We could classify each of its input and output channels connecting with an object in that domain in terms of the channel's rate of flow (null, low, or high) and its pattern (one-shot, intermittent, or steady), its level (physical, symbolic, or rational interaction) and its modality (physical level modalities: mechanical, chemical, optical, sound, etc; symbolic level modalities: written language, spoken language, pictures, notes, gestures, etc; rational level modalities: ask, answer, suggest, promise, argue, encourage, entertain, etc).

Object types in the interaction domain could be things, humans, teams, organizations. We could make a finer division into roles, for instance in the case of a human environment, perhaps controller, user, and client or consumer. Even with as few as three objects in the interaction domain, this will give us an umpteen-dimensional table with many boxes not yet filled in by some existing computer application.

Or maybe we should try with a categorization based on functionality. But if we already have decided on the permissible functions, how are we ever going to come up with something new? Put a symbolic interface onto anything, and then the power of the computer can be applied to it. Its power is to relate anything to anything else in any conceivable way. (Almost.) What can we conceive? Perhaps it would be better to start by making a long list of conceivable and inconceivable computer applications: automata, tools, toys, possibility explorers, collaborators, guardian angels, media, artificial worlds ("Through the Computer-Screen. Everyman's Adventures in Wonderland").

What do you do with (in) artificial subworlds created in a computer? Have fun, learn things about the real (?) world or possible outgrowths of it, learn things about yourself, develop new abilities, create works of art, adventure. Or you sink into the artificial world making it the real world. Change your old life to a brand new of your own choice, put yourself in a life-support system, enter your own world and earn your living (that is pay for the life-support system and computer power) with some symbolic, intellectual work you can do within your new world.

If you're not satisfied with making the new worlds by symbols alone, let's make them by hands, supported in our worldmaking by possibility explorers, computer collaborators, design environments, industrial robots, and computerized guardian angels. We could start with small things, smart objects like radios and teves, cars, houses, books. Why not a smart desk? Combine the electronic desktop with a solid one by putting identification tags on all items.

We can deconstruct the mechanical world and reconstruct it with symbolic manipulations between the parts. Simulate physics not of this world. Give null-inertia to heavy motor parts, hold a ship steady as a rock in full storm. We can impose arbitrary relations, that can be changed according to any parameters, increasing flexibility, checking constraints, make the world a safer place. A sudden full turn of the steering wheel at high speed on the highway does not mean death, if it's a symbolic turn.

Or we could make something more ambitious, something that involves people in a deeper way, like new ways of using an electronic message system to get things done in a group or a team. How about a new group decision technique, somewhat like the Delphi method, with a point moving between decision alternatives in the force-fields of mouse-clicking decision makers? Can we deconstruct and reconstruct social institutions analogous to reconstructions of the mechanical world? Isn't that what system designers have been doing all the time? Now that we realize what we have been doing, could we use the technique for something better?

3. An invitation to creative conversation #2

Computer technology can be used for data storage, but the computer certainly is no *data* machine, its use is not restricted to information system support. But then again, the computer is not really a data *machine* either. Or better, to think of the computer as a machine demands a rather sophisticated notion of what a machine is. A machine is not a clock.

Just as the clock played a central role in the 17th century development of the mechanistic world view, so one could try to formulate a world view on the basis of the computer. What would such a world view look like, and how would it differ from the mechanistic world view?

It is common practice in systems development to identify basic assumptions expressing different perspectives on the use of computer technology. A standard categorization is that of Nurminen (1988), that we have looked at above, into three such perspectives. We prefer to think of Nurminen's three perspectives—the systems-theoretical, the socio-technical, and the humanistic perspective—as each exemplifying a more fundamental world view: a mechanistic, an organic, and a traditional. You can say interesting things about computer technology in these terms, but what you then do is try to understand this *new* technology in terms of categories developed for old technologies.

The discussion of technology has since the early 19th century at least, put up these three world views against one another. What we would suggest a *creative* conversation could do is to attempt to develop new perspectives, ideas for new world views based on the new technology, and then maybe go on to discuss those ideas.

Since technologies influence our ways of thinking, let us ask ourselves what ideas this new technology will give us, will encourage, etc. And then, let us go on to ask if these are good or bad ideas, if they are humanistic or degrading, oppressive or liberating, etc. And let us consider what should be said and done in order to push these ideas, within their degrees of freedom, in good directions.

There are ample opportunities to study the effects on ideologies and culture of earlier technologies. It is more difficult to determine the role of such technology-inspired ideas in changing society. An example from Mumford (1934) illustrates these difficulties. Mumford argues that mining technology was used as an exemplary model when the factories and factory work were designed, and when railways and subways were built. He also claims that the power of the mining technology as a model was enhanced by the fact that the new physics described a colorless reality, very much like the world of a mine. One wonders if this ideological detour had any importance, and one wonders how the mining technology could become such a powerful model. Why was it not met with more stubborn resistance?

It is even more difficult, today, to comment on the possible effects of computer technology on our new world views, and the practical implications of such world views. But that is just what we want to do in this our second conversation. Now we won't look, as we did in our first conversation, at the concrete details of computer technology in the near future. We'll look at its potential, in a longer perspective, to change our ideas about society, man, work, and at the possible consequences such new world views can have on the details of social life.

We are not interested in prediction. We believe that there is an important relation between what people today *think* computer technology is, and our future use of this technology. When we become conscious of our preconceptions about the computer technology it becomes possible to liberate ourselves from them and consider other ways of conceiving this technology. We are not the fateful victims of a historically determined computer technological ideology. But unless we consciously work on the preconceptions we have, making efforts to discover and consider alternative conceptions, and their consequences, it will look as if that was the case. We need to keep many possibilities in the air in this period of rapid diversification of the use of computer technology.

Old ideology still rests its heavy hand upon computer technology. Still, more than forty years since the birth of the computer, we choose between viewing the computer from a pre-industrial tool perspective and as a sort of clockwork ticking zeros and ones rather than cogs. We go on thinking and talking of the computer as a "data machine," of the typical computer application as an "information system." We think it is necessary today, to try to develop many different new, untested perspectives, and consider the consequences of these different viewpoints. And we want your help.

Here we shall only exemplify with one, fairly coherent, way of looking at computer technology and its use. It is based on what we think is a particularly important novelty in a budding computerized world view, namely the fact that the given, unique, necessary world of the old mechanistic world view, is being replaced by a constructive, ambiguous world of possibilities. Again, the sketch we give below is only intended as something to fall back on when creativity fails:

come that sad moment we can always turn our conversation into a criticism of the following sketch of a computer based world view.

4. A computer based world view

We tried to sketch seven elements of a computer based world view in Dahlbom & Janlert (1988). Computer technology, as we saw it then, invites us to a world that is artificial, external, cognitive, simulated, designed, interactive, and parallel. This sketch can certainly be developed and questioned, and more elements can be added. Let us give a summary of what we then wrote about these elements:

Our artifacts are shaped by our experiences. It is but a small step to the conscious use of artifacts to support memory, problem-solving and planning. Thinking is then "externalized" in artifacts. Language, mathematics, methods of book-keeping are examples of intellectual artifacts which expand our power of thinking by subjecting thought to external, socially controlled systems of rules. As far as thinking consists in the manipulation of symbols in such systems, it is "artificial" rather than "natural" (Dahlbom 1989a has a little more to say on this).

These intellectual artifacts, systems of rules, are materialized in physical objects of various types. Intellectual work is becoming increasingly dependent on these concrete artifacts. In a society without books you must keep all your knowledge in your head. With books as secondary storage you can handle a vastly greater amount of knowledge, but now priorities change with regard to how your primary storage, your brain, should best be used. You need to handle your secondary mass storage expertly. You need fast reading and writing, skills in finding information, in reasoning, analyzing, synthesizing and communicating. This externalization is amplified by computer technology. As our knowledge grows, more and more of it is external to us, and our internal knowledge will be *about* knowledge, what there is, how to find it, how to apply it, etc. The substance of what we know is no longer inside us, but in the networks of men and computers we belong to.

This externalization changes our view of our selves and of what it means to know oneself. When a person's identity is made up of external artifacts she can only learn to know herself through these artifacts and they are as easily accessible to other people. The self is no longer private. It becomes public and objective, social or cultural rather than natural.

A computer based world view puts great weight on knowledge. But not just any kind of knowledge. It is only the Cartesian kind, representing or mirroring the world, that counts. In perceiving and thinking about the world we are manipulating symbols that represent the world, rather than the world itself. The symbols are not natural images but artifacts we've made which represent by virtue of our conventions. When we realize that "the limits of our language mean the limits of our world" our quest for knowledge becomes dependent on our ability to construct artificial symbol systems.

Artificialization leads to a view of knowledge where what can be known coincides with what can be constructed. Computer technology makes it possible to automatize the construction. Computers enable us to study processes by simulating them. They make it possible to study not just the actual processes but also possible processes. Computer technology transforms all intellectual work into design, construction.

As automata for simulation, the role of computers will be more active than that of traditional intellectual artifacts. In the same way that we obtain information of the world by interacting with it, our computerized design experiments will involve interaction. We will be engaged in a multitude of such experiments in a world filled with computer artifacts interacting with us. We will live parallel lives in different possible worlds.

The most revolutionary change in our world view brought upon us by computer technology, is perhaps its reconstruction of space and time, the fundamental categories of the mechanistic world view, and of a modern capitalistic, industrialized society. Our movements in space will be individualized and eclipsed by movements in computerized memory spaces. Our need for local and personal coordinations will be supported by computers, at the same time as the need for central synchronization becomes less, and with that goes our interest in a common, standard time.

In mechanistic space I can never be in two places at the same time, in mechanistic time I have to do things in sequence. In computer space I can be in many places at the same time, in computer time I constantly do things in parallel, and skip freely backwards and forwards in time. A symptomatic, simple, example is the popularity of video-taping teve programs to get control over time, get rid of commercials, boring parts, etc.

Together these elements of a computer based world view give us a picture of a world where man is moving further away from nature, further out into a world of artifacts.

5. An invitation to creative conversation #3

System development is still generally conceived as the "design and development of information systems." System developers go on thinking of their practice in the very terms used in the 60's when information technology was used in administrations as information systems. But so much has happened since then, in the production of computer applications, that it is high time, we think, to reconsider the practice of system development in view of a more general and diversified notion of information technology use. In a cycle of innovative use the defining competence of a system developer lies in the area of "technology use."

Even if it is possible to think of design tools, computerized tools (from golf clubs to chain saws), word processing systems, calculation systems, communication systems, etc, as "information systems," it is not very informative. The difference between applications like these, which were not available in the 60's, and the kinds of systems then being introduced—client registers, salary systems, inventory systems, book-keeping systems, etc—are deep-going enough to warrant a whole new conception of what system development is all about.

As long as we think of the standard computer application as a huge computerized information system, we will worry about deskilling, alienation, loss of liberty and flexibility, dreary and meaningless work tasks, and other forms of dehumanization. Our natural alternative will be a return to pre-industrialized, individual tools. The background for our moral standpoints will be the computer as an element in a mechanistic world view, and Marx' and Braverman's analysis of the consequences of such technology. But once we start thinking about future computer technology use along totally different lines, once we see that technology as part of a whole new world view, we shall have to reconsider our moral obligations.

What will be the role of future system developers, and what will be their moral problems? Where will they be employed, what will they do, what will be their specific competence? These are questions for the creative conversation #3. Again, we supply some tentative answers (drawn from Dahlbom 1989b), which certainly aren't by far creative enough compared to what we hope will come out of our conversations, but which may serve as some sort of background when we run out of steam.

6. System development in the future

Our understanding of what system development should be, ought to change with the changes in information technology and its use. Practitioners have to adjust their practice to these changes, and as researchers we have to pay close attention to changes in the practice of system development. This practice is currently undergoing a process of industrialization. From having had the character of consulting in short-term projects, at the time of buying, installing and breaking-in of personnel and machines, system work is changing into being (1) a permanent part of maintenance, (2) a part of the software industry, (3) done by end-users, (4) product information, and (5) general, non-technical organizational development.

To our mind, Scandinavian research on system development has been much too slow in its reaction to these changes. The discussion of system development practice is still dominated by a conception of it as "project-work," and major efforts are spent on the details of "project-design." (A good example is Bjerknes 1989.) But there will soon be no place for detailed system analysis and design work in the field. The typical "system designer" will be assembling ready-made products. And those ready-made systems will be developed at a rate making projects impossible. Like in the UTOPIA project, the desired tool will be developed by industry while the project is still playing around with cardboard prototypes (Ehn 1988).

After having been fully occupied with the engineering aspects of the technology, system development research began, in the seventies, to pay more attention to the changes in work situations brought about by the new technology. From having been a discipline for engineers, it became more of a discipline for (engineers turned) social scientists and philosophers. Both these approaches to system development research had a relatively unimaginative notion of the possible varieties of information technology use. The engineers had their hands full with file handling and database management, the philosophers with tacit knowledge, democracy and the dequalification of work. In the meantime a revolution began in the area of information technology use.

The growing supply of factory-made software and standard systems made the system consultant as craftsman—engineer in a pioneering field—the victim of industrialization. The increasing variety and complexity of use made it less and less interesting to discuss the consequences of information technology in general. The two research approaches that were established to support the practice of system development had to adjust to these changes as the engineering consultants were entering software laboratories and the organization-oriented consultants became advanced sales persons or general consultants. So, the research approaches had to change orientation, and they did so by moving in the direction of (2) and (5) above, thus moving further apart.

But all that is changing now that more and more of us begin to realize that the exciting future of information technology research lies in questions of use. The field of "human-computer interaction" understood in a most general sense, covering everything from the research approach so called to questions of how to live with computers, is bridging the gap between engineering and social consequences (Janlert 1989). What a future system developer will need most of all is a rich picture of the possibilities and varieties of human-computer interaction. She'll be working with a plethora of hardware and software on one side and with users and user organizations on the other, and it will be her job to make the twain meet. She'll be an expert on what to do with all the artifacts produced by computer industries, rather than on how to produce those artifacts. Or is that too boring?

References

Bjerknes, G. (1989) Motsigelses-begrepet—et redskap for å forstå situasjoner i systemutvikling. Institutt for Informatikk. Universitetet i Oslo.

Dahlbom, B. (1989a) Artificiell och naturlig intelligens. In N.O. Finneman (ed.) *Tidens tegn. Natur—Information—Kultur. Kulturstudier 3.* Aarhus Universitetsforlag.

Dahlbom, B. (1989b) Using technology to understand organizations. Forthcoming in Bjerknes et al (eds.) *Organizational Competence in System Development.* Lund: Studentlitteratur.

Dahlbom, B. (1990) The Idea that Reality is Socially Constructed. Forthcoming in R. Budde et al *Software Development and Reality Construction.* Berlin: Springer-Verlag.

Dahlbom, B. & Janlert L.-E. (1988) *En artificiell värld.* MDA-rapport. Arbetsmiljöfonden och Styrelsen för teknisk utveckling.

Ehn, P. (1988) *Work-Oriented Design of Computer Artifacts.* Stockholm: Arbetslivscentrum.

Janlert, L.-E. (1987) The Computer as a Person. *Journal for the Theory of Social Behaviour,* 17: 321-341.

Janlert, L.-E. (1989) Models in Human Computer Interaction. Report UMINF-161.89. Umeå Universitet.

Mumford, L. (1934) *Technics and Civilization.* New York: Harcourt Brace Jovanovich.

Nurminen, M. I. (1988) *People or Computers: Three Ways of Looking at Information Systems.* Lund: Studentlitteratur.

Rorty, R. (1980) *Philosophy and the Mirror of Nature.* Oxford: Basil Blackwell.

Zetterberg, S. et al (1984) *Det osynliga kontraktet. En studie i 80-talets arbetsliv.* Stockholm: Sifo.

PLAYING IN REALITY

Pelle Ehn
Department of Information and Media Science
University of Aarhus
DK-8200 Århus N
Denmark

Bengt Mölleryd and Dan Sjögren
Swedish Agency for Administrative Development
Box 34107
S-10026 Stockholm
Sweden

Abstract: This is a paper about "design-by-playing" and an attempt to take *homo ludens* seriously. We will illustrate this approach by a game played at *Konsumentverket*, the national Swedish board for consumer policies, in 1988. The main reason for us to construct such an organisational design game was the shortcomings we had encountered in our earlier use of traditional systems description methods in participative design processes. No matter how suitable the traditional methods may have been for making requirement specifications for technical implementation, they did not support participative and involved acting. Hence, the move away from the safe ground of correct systems descriptions to the open-ended games of participative action - from systems descriptions to scripts for action.

The dramatic design context the game offered at Konsumentverket was based on six concepts. The *playground* is a subjective and negotiated interpretation of the work organisation in question. The *professional roles* are the union of individual professional ambitions and the need for qualifications from an organisational perspective. The *situation cards* introduce prototypical examples of breakdown situations. *Commitments* are made by individual role players as actions related to a situation card. *Conditions* for these commitments are negotiated. The game starts with a prologue were the rules of the game are introduced, the proposed playground is redesigned, and the role cast is chosen and ends with a final act in which an *action plan* for negotiations with the surrounding organisation is formulated.

As a background for the seriousness of the "design-by-playing" approach a theoretical framework grounded in the language-games philosophy of the later Wittgenstein is outlined.

Key words: systems description, design, organisation, change, game, play, metaphor, script, role, commitment.

1 Introduction

"You are at a restaurant. At a corner table you happen to see your best friends wife/husband in an intimate scene with another man/woman. Would you tell your best friend what you have seen?"

This could be a situation card which you have drawn in the family game *Scruples*. According to the rules of the game you are supposed to answer: "yes", "no" or "may be". The other participants will then judge if they believe you. You are allowed to argue for your answer. These are some of the *explicit* rules of this family game on moral and ethics. But playing the game also involve following *unwritten* rules. If your best friend and his/her wife/husband are participating in the game your answer and arguments may be different from if they were not present. Here it is your social competence rather than your knowledge about the explicit rules of the family game Scruples that helps you to make a socially acceptable *move* in the game. In playing this family game you may even decide to change the rules. You may for example agree with each other that the answer "may be" is forbidden, or that you can refuse to answer a question if you find it boring. You change the rules as you play along.

To us this game situation has a *family resemblance* with the social activity of systems development or design of computer artifacts. This is (1) a normative, (2) a methodological, and (3) a theoretical position we take in this paper.

1. As a normative point of departure we suggest that the idea of design as *playful engagement* may be a creative way of transcending the boredom of traditional design. We strongly favour the idea of participative design, and take the position that *successful participation requires that design is meaningful to all participants and preferably pleasurable to them as well.* We suggest that *homo ludens* should be taken seriously, playing the main role that earlier has been dedicated to *homo sapiens* and *homo faber*.

2. Methodologically we suggest that *game and play metaphors* may be a creative way of introducing the design process to users, and for organizing participative design. In design methods *linguistic artifacts to support social interaction and creativity* come into focus rather than e.g. construction of correct data flows based on interviews of data.

3. Theoretically we find inspiration for our position in the ordinary language philosophy by Ludwig Wittgenstein, and the shift of focus from language as description towards *language as action*. Concepts like *language-games, family resemblance, rules, moves, speech acts, linguistic and other artifacts, paradigm cases* and *reminders* rather than systems, data, information and descriptions become central in our way of doing design and in our detached reflections on this activity.

In this paper these ideas of *design-by-playing* are discussed with reference to a *paradigm case* we carried out at the national Swedish board for consumer policies *Konsumentverket* during the fall of 1988. However, before going into details of the design game played in this case we will relate our design-by-playing approach to approaches with *family resemblance*. But first a few words of theory on playing the language-games of design an use.

2 Playing the Language-Games of Design and Use

2.1 *Language-games*

To use language is to participate in *language-games*, the Wittgensteinian notion of practice [Wittgenstein 1953], [Ehn 1988]. In discussing how we in practice follow (and sometimes break) rules as a social activity Wittgenstein asks us to think of games, how they are made up and played. Why games?

We often think of games in terms of *playful, pleasurable engagement*. As mentioned above we do not think that this aspect should be denied, but a more important aspect of the games children play is that these games are *most concerned activities*, as are most of the common language-games we play in our ordinary language. Even professional language-games of e.g. designers, carpenters or typographers, complicated as they may be, are grounded in our everyday ordinary language.

We do not understand what counts as a game because we have an explicit definition, but because we are already familiar with other games. There is a kind of *family resemblance* between games. Similarly, professional language-games can be learned and understood because of their family resemblance with other language-games which we know how to play.

Language-games, like the games we play as children, are social activities. To be able to play these games we have to learn to follow rules, rules that are socially created, but far from always explicitly existing. The rule-following behaviour of being able to play *together with others* is more fundamental to a game than explicit regulative rules. Playing is interaction and cooperation. It is inter-subjective practice. To follow the rules in practice means to be able to act in a way that others in the game can understand. These rules are "embedded" in a given practice from which they cannot be distinguished. They are this practice. To know them is to "embody" them, to be able to practically apply them to a principally open class of cases.

Language-games are performed both as speech acts and as other activities, as practice with "embodied" meaning within societal and cultural institutional frameworks. This seems to make us prisoners of language and tradition, which is not really the case. Being socially created, the rules of language-games, as those of other games, can also be altered. There are, according to Wittgenstein, even games in which we *make up and alter the rules according to which we play, as we go along*. The language-games of design are a good examples. Actually, the *creative transcendence of traditional behaviour* is what is typical for skilful human behaviour, and exactly what defies formalization. By mastery of the rules comes the freedom to extend them. This creativity depends on the Wittgensteinian emphasis on the open-textured character of human rule-following behaviour. To begin with, we learn to follow a rule as a kind of dressage, but in the end we do it as a creative activity. To be able to follow a rule is to have learned how to in practice continue an example we have been given. Mastery of the rules put us in position to invent new ways of carrying on. There is always the possibility that we can follow a rule in a wholly unforeseen way, and still it will be accepted by others as a correct move.

The idea of language-games entails and emphasis how we linguistically discover and construct our world. However, language is understood as our use of it, as our social, historic, and inter-subjective application of linguistic artifacts. In this paper we focus on the use of such linguistic artifacts in design games. However, as we see it, this is not a neglect of how we as humans also come to understand the world by use of other artifacts. Artifacts and objects also play a fundamental role in a given language-game. A hammer is in itself a sign of what you can do with it in a certain language-game. And so is a computer artifact. This is for example why "what-you-see-is-what-you-get" and "direct manipulation" interfaces are important. These signs remind you of what you can do with it.

2.2 From description to reminder

In a Wittgensteinian approach, focus is not on the "correctness" of systems descriptions in design, on how well they mirror the desires in the mind of the users, or on how "correctly" they describe existing and future artifacts and their use. Systems descriptions are *design artifacts*, typically linguistic artifacts. The crucial question is how we use them, what role they play in a design language-game. The new orientation suggested in a Wittgensteinian approach is that we see these linguistic artifacts as a special kind of artifacts that we refer to as "typical examples" or "paradigm cases" when we describe something, or when we "inform" each other. In the language-game of design we use these artifacts as *reminders* and as *paradigm cases* for our reflections on future computer artifacts and their use. The use of design artifacts brings earlier experiences to our mind and it "bends" our way of thinking of the past and the future. This is how we should understand them as *re*presentations. And this is how they "inform" our practice. If they are good design artifacts, they support good moves within a specific design language-game.

2.3 Creating design language-games

As designers we are involved in reforming practice, in our case typically computer artifacts and the way people use them. Hence, the language-games of design changes the rules for other language-games - those of use of the artifacts. What are the conditions for this interplay and change? What do we as designers have to do to qualify as participants in the language-games of the users? What do users have to learn to qualify as participants in the language-game of design? And which means can we develop in design to facilitate these learning processes?

To possess the competence required to participate in a language-game requires a lot of learning within that practice. But in the beginning all you can understand, is what you have already understood in another language-game. You understand because of the family resemblance between the two language-games. However, by understanding design as a process of *creating new language-games* that have family resemblance with the language-games of both users and designers we have an orientation for really doing design as skill based participation. This approach offers a way of playing and doing design that may help us to transcend some of the limits of descriptions, systems and formalization. Our paradigm case of a desktop publishing game in this paper we refer to as such a new design language-game. This language-game has family resemblance with our own earlier design efforts and with design approaches developed by others.

3. Design Language-Games with Family Resemblance

3.1 Games we played before

The basic ideas behind the design game presented in this paper go back to to our experiences from two research projects carried out in the first part of the 1980s. The UTOPIA project concerned the design of tools for skill enhancement in the high tech domain of newspaper pre-press production [Ehn 1988]. The Carpentry Shop project concerned technological and organizational alternatives in the "low tech" domain of small and medium scale carpentry shop production [Ehn and Sjögren 1986]. Though the application domains and level of technology were very different in the two projects there were many common features.

Some of the main features were the participative design approach and the understanding of the design process as a process of mutual learning between professional designers and skilled users within the application domain, and as a process were future or alternative technology and work organization were envisioned and experienced rather than described. Aspects common to the design approaches used were a focus on concreteness and ease of use. The design approaches included mock-up simulations, prototyping, organizational games supporting investigation work and designs in study circles or other user discussions and design groups. The use of mock-ups and prototyping as well as the organizational design games were a reaction to the shortcomings we had encountered in our earlier use of traditional systems description methods in a participative design process. No matter how suitable the traditional methods may have been for making requirement specifications for technical implementation, they did not support participative and involved acting. Hence, the move away from the safe ground of correct systems descriptions to the open-ended games of participative action.

The use of mock-ups and of prototypes opens up possibilities for "design-by-doing" - for getting hands-on experiences with future technological alternatives. In this paper we focus on the use of organizational games and how they may open up for "design-by-playing" - for getting involved design discussions also of the overall work organization, skill requirements, division of labour and cooperation in the work process.

The basic ideas behind the organizational design games in UTOPIA and the Carpentry Shop Project were:

• they should be fast and easy for a group of people to work with,

• they should be cheap and flexible to use allowing several alternatives to be tested during discussions, and

• they should be based on concepts relevant to the actual type of production and support design discussions of existing and future work and technology.

The design tools used in the games consisted of boxes of cards, each card representing a work or a machine function. The cards were used to build up large layouts of the organizations in different carpentry shops or newspapers. These layouts could easily be changed and alternative organizational design could be mapped out quick and cheap.

The design games in both projects were built up around a *production flow metaphor*. However, differences in type of production, especially the level of concretion, were reflected in the design games. In carpentry production the machine-tools are still tangible and visible and an icon is a good reminder of the work activity. The layouts become a physical type of reminder of a factory or a shop. For pre-press newspapers production the functions of the machines are no longer quite visible and tangible. Icons of computers do not express very much of what is going on. The design game in this case was more oriented towards work functions represented by icons.

The carpentry layout design game has been used in the development work to outline demands on quality of work and products. In education the game was used as part of a trade union strategy by workers at individual shops to discuss the present organization and to work out alternatives. The work with the layout design game was complemented by playing the "Carpentrypoly Game" - a game of *Monopoly* type for experiments with different business strategies on investment policy and work qualifications.

The newspaper organization design game has been most suitable for vocational education and in preparations for local and central negotiations on "demarcation problems" and in connection with purchase and use of new technology (and organizational changes in general).

3.1 Other design games

The experiences from the UTOPIA project and the Carpentry Shop Project point in a direction away from the main road in systems development of understanding systems design as a matter of constructing true pictures of reality. Instead, a direction towards understanding design as a game of intervention into practice is suggested. This new orientation is by no means exceptional. It can e.g. be found both in the "soft system methodologies" and in more linguistically oriented approaches.

Russell Ackoff as a representative of the "social systems approaches" has concluded that for participation in design to be successful it requires that:

- it makes a difference for the participants
- implementation of the results are likely
- it is fun to participate.

The two first points concern the political side of participation in design, the users must have a guarantee that their design efforts are taken seriously. The last point concerns the design process. No matter how much influence participation may give, it has to transcend the boredom of traditional design meetings to really support design as meaningful and involved action. According to Ackoff the principles of *idealized design* [Ackoff 1974] leads to a design process that is both liberating and fun. The design work is treated as a play. The approach we have taken have family resemblance with participative idealized design.

But it also resembles some of the features of Peter Checkland's Soft Systems Methodology [Checkland 1981]. Hence the syntax and semantics of our design games are as in the case of Checkland 's *rich pictures* situation dependant and open for changes all the time. Furthermore, the linguistic design artifacts being used in our design games resemble Checkland´s understanding of his *conceptual models* as systemic artifacts for reflection and action. If the word system has a meaning it is as systemic human activity.

As a final example of approaches with family resemblance we will mention the one taken by Winograd and Flores and their theory of language as social action [Winograd and Flores 1986]. They think of design in terms of fundamental linguistic actions, i.e. requests, promises, assertions and declarations. The idea of design as a linguistic game in which the participants make linguistic actions and commit themselves to future actions is also a basic idea in our design game. However, as outlined above, our understanding of language as action is based on a more general interpretation of the Wittgensteinian ordinary language philosophy.

4 Situating the Desktop Publishing Game

The design approach we present in this paper we will refer to as the *Desktop Publishing Game*. This design game means a shift of environment from the shop floors of the Carpentry Shop and UTOPIA projects to work in an *office* environment. Certainly many different activities are going on in offices. We have chosen to concentrate on changes related to the introduction of new technology for publication work, on *desktop publishing*. One obvious reason for this is the fact that we had some earlier experiences within a domain with family resemblance -

typographic work. Another reason is the vast investments in this technology in offices today, and the increasing demands on communicative competence on individuals and organizations.

If we compare the Desktop Publishing Game with the design games from UTOPIA and the Carpentry shop project there are not only similarities, but also clear changes. The participative language as action approach is still fundamental. Concreteness and ease of use are still basic principles. However, the flow oriented production metaphor used in the earlier games did not make sense in the office environment of desktop publishing. Hence, the shift of metaphor towards a game based on linguistic actions leading to commitments. Social interaction and cooperation come into focus rather than the physical artifacts used [Wynn 1979]. Furthermore, the play metaphor is used much more explicit and the theoretical motivation for this is more elaborated.

Another difference that is striking is the design focus. Existing hardware and software are more or less taken for granted. This is based on the assumption that the basic problem in the domain not is technology driven, but a question of organizational change, education and re-qualification. The technology is good enough. The way the game is designed it takes as point of departure that the users are experts on their situated work, but that the professional designers as well have relevant and useful competence in the application domain. Hence, in this specific design game it is suggested that design basically concerns organizational design and not software development, and that the professional designers to some extent know more about the application domain then the participating users. Neither of these assumptions seam at first to be justifiable compared with earlier experiences and basic assumptions. By presenting the game we hope to be able to demonstrate that these are relevant moves we have done in establishing the Desktop Publishing Game.

The Desktop Publishing Game make use of tools for making *graphical reminders* like notice boards, cards and personal scripts. It is *language* oriented in its use of professional and situated concepts from the tradition and the application domain. It is *concrete* in its use of critical situations and events in the actual organization. It is *dramatic* not only in its use of the game metaphor, but more specifically in its use of a theatrical play metaphor.

We have played the Desktop Publishing Game in a few offices in the public administration in Sweden. However, our main case is the consumers agency *Konsumentverket*. In the work with information for consumers the agency has a educational department. The 25 employees at the department produce books, reports, educational material, and brochures. The background for the project using the Desktop Publishing Game was the following. The department had invested in desktop publishing technology. However, the lack of competence to master this technology caused uncertainty about professional roles and inability to raise the typographical quality. To really use the technology required development of the professional roles, changed professional relations, education and a redesign of the work organization. In this situation the management of Konsumentverket contacted The National Fund for Administrative Development and Training for Government Employees and asked for support. This is how we entered the stage in a project financed by the governmental initiative for development and education in the public sector, and this is how user participation was financed as well.

Figure 1. Design-by-playing at Konsumentverket, September 1988.

5 The Design Game at Konsumentverket

In this section we will introduce the game as it was played at Konsumentverket. The game was played in four acts. A prologue of half a day setting the stage. A first act of two full days in which relevant situations are simulated and commitments are made. A second act of two more days in which earlier real publications are "replayed" in the light of understanding from the first act. In the third act the last two days are spent on transforming conditions and commitments from the game into an action plan for redesign of work and use of technology.

5.1 Setting the stage

At the first meeting between us (the designers) and the participants from Konsumentverket we introduced the ideas of the game as a way of developing professional roles and the work organization. The basic rules of the game were explained using examples from another case. The participants from Konsumentverket gave examples of problems they have in their practice and showed examples of products they had made. This was the first step in establishing a common language-game. Together with us this language-game was played by eight employees at the education department of Konsumentverket. They were secretaries, case handlers, a technical support person, a marketing person, and the head of the department. As resource persons an information consultant and a typographer also participated occasionally.

5.2 The prologue (4 hours)

When the stage was set we started to work on the *prologue* which meant preparing the *playground* and the *cast.*

The *playground* was a crude reminder of work situations that publication work normally consist of. It consisted of twelve sheets of paper each referring to a specific work situation. These situations ranged from publication ideas to marketing. The situations were taken from typographical as well as editorial professional traditions.

At the prologue meeting the playground was put on the wall. We discussed the work situations and how they were labelled. The playground was revised and clarified until it worked as a reminder for the participants situated experience and consensus was reached on the interpretation. For example the proposed situation "investigation" was replaced by "gathering of facts".

We had prepared the *cast* was prepared for traditional roles like, the editor, the executive editor, the graphic designer, and the author. But there was also new roles like, the technology coordinator, the publication administrator, and the "formatting person". These are extensions of the secretary role. The roles were partly taken from the situated practice at Konsumentverket and partly from roles we had met or created in an earlier case. To each role a *role script* was prepared.

At the prologue meeting, however, also the cast was revised. For example the role of a "picture editor" was suggested by the participants to be integrated into the role of the editor, and a marketing role was added after discussion. The role scripts were distributed and studied by the participants who choose at least one role each. The roles were chosen according to realistic professional ambitions of the participants and organizational demands. For example a secretary choose to play the role of the "formatting person" and another choose the role of the publication administrator.

The actors prepared for *Act I* by studying and revising their role scripts, and by writing *situation cards* for the plot.

Each role script consists of the following parts: qualifications, tools, and work situations to participate in. Qualifications had a crude default pattern. This is what was refined in the preparation.

The *situation cards* are what drives the plot forward. Each card reminds of a critical event, a breakdown situation or a daily triviality which sooner or later is likely to occur. There are three types of situation cards: paradigm cases from other organizations using desktop publishing technology, role oriented cards, and cards specific for the actual organization. The organization specific cards on problems at the education department were the cards the actors prepared. The other cards were prepared by the designers.

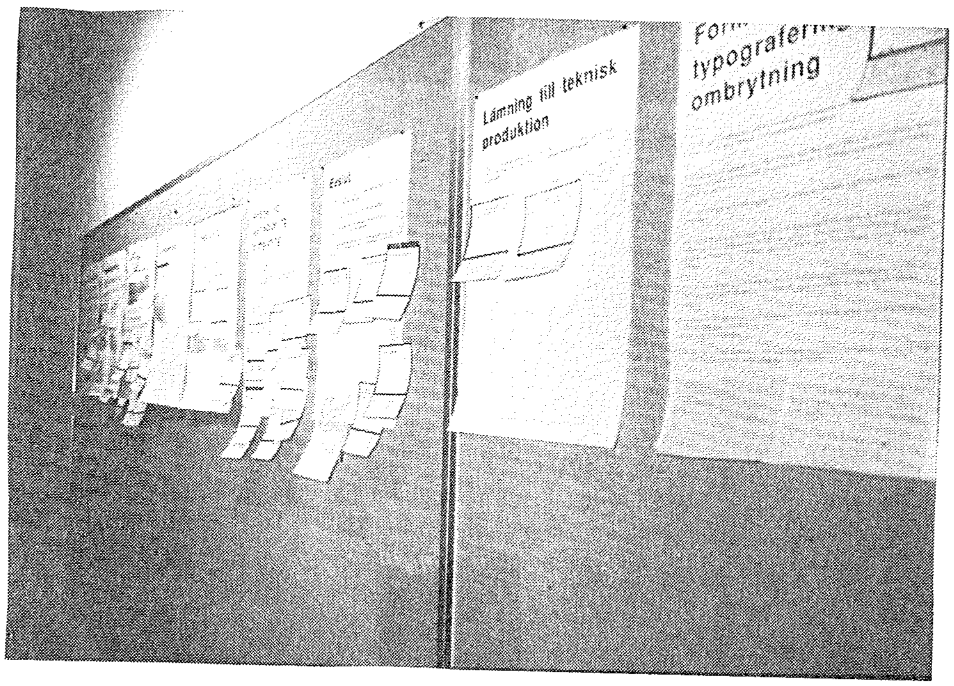


Figure 2. A playground including the work situations from "the idea of a publication" to "the marketing of a publication".

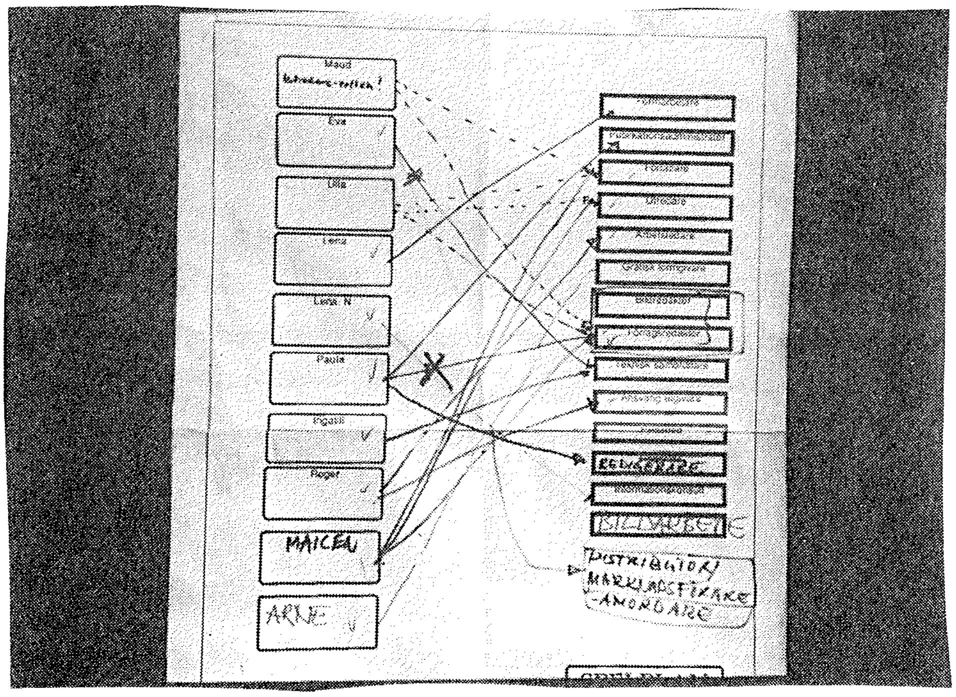
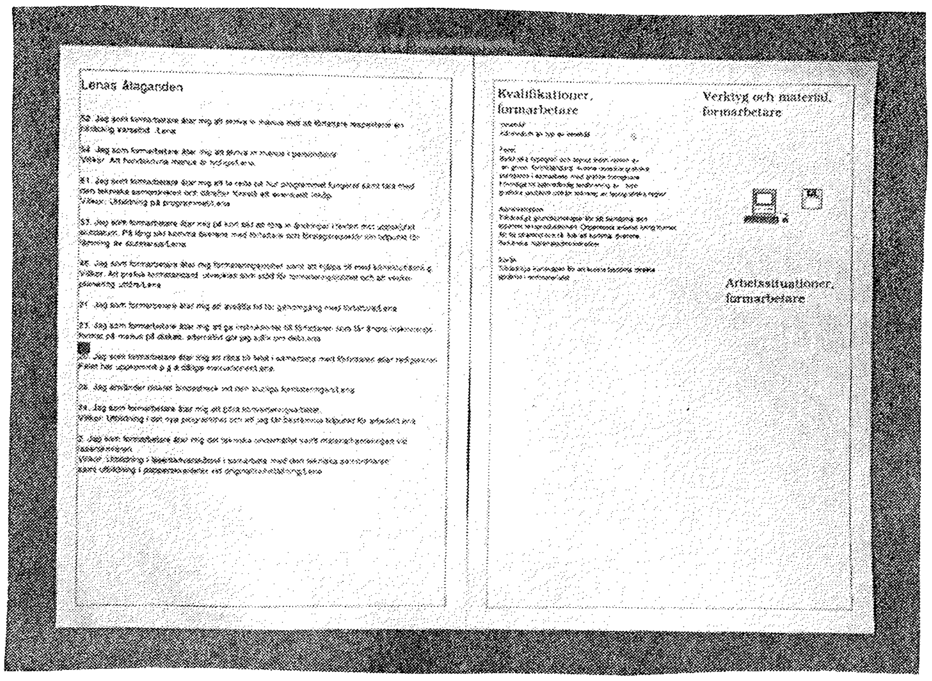


Figure 3. The role cast prepared and proposed by the designers and revised and changed by the actors.



Lenas åtaganden

[illegible]

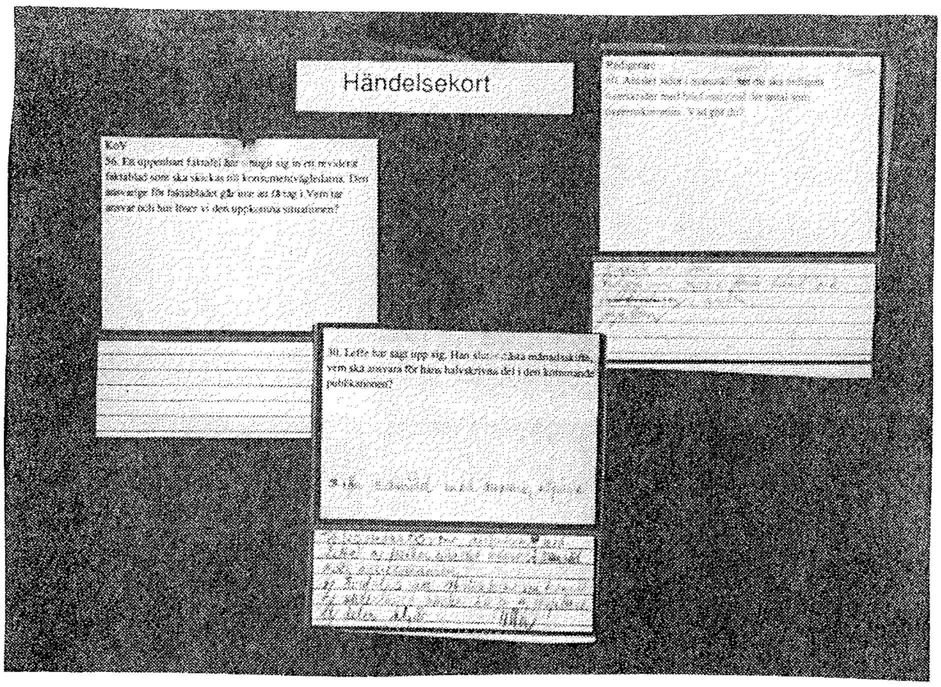
Kvalifikationer, formarbetare

[illegible]

Verktyg och material, formarbetare

Arbetssituationer, formarbetare

Figure 4. A role script containing commitments of an individual actor and his/her conditions in terms of qualifications, tools and work relations.



Händelsekort

KoV
56. En uppenbart felaktig [illegible] smugit sig in ett reviderat faktablad som ska skickas till konsumentvägledarna. Den ansvarige för faktabladet går inte att få tag i. Vem tar ansvar och hur löser vi den uppkomna situationen?

30. Leffe har sagt upp sig. Han skulle [illegible] månadsskriften, vem ska ansvara för hans halvskrivna del i den kommande publikationen?

[illegible]

Figure 5. Situation cards. Each card describes a problematic situation brought in by the designers or a"breakdown situation" from the players own experience.

5.3 Act I (2 x 8 hours)

The actors meet with the designers on the stage. The playground is placed on the wall. 81 situation cards have been made and are put in a stack on the table. The actors have put their individual role scripts on the table. The designers give a short introduction to Act I. The play can begin.

One of the participants picks a card. It reads: "Printing of camera ready original on laser printer is still in progress. It must be delivered to the print shop before 4 p.m. today. The toner cartridge is suddenly empty, and no one has bothered to order new ones. Who takes responsibility to solve the problem now? And what should be done to avoid similar breakdowns in the future?" This certainly reminds the participants of a familiar situation that typically "falls between the chairs". After a discussion the secretary that had chosen the role of the publication administrator makes a *commitment.*

This commitment means that she takes the responsibility to solve the problem under certain *conditions.* For example she may state: "In my role as publication administrator I commit my self to solve the immediate problem by ordering a new cartridge and have it sent by taxi. In the long run I will establish a routine for the maintenance of the laser printer. My condition is that I get a appropriate training in maintenance of the printer." The group actively support her commitment. The commitment is written down on the situation card. The card is placed on the playground in the work situation labelled "printing". She make a similar note in her own role script.

The game continues with more cards in the same way. The group at Konsumentverket got on well and had no problems with the cards even if some situations were really tricky. However, on some situations the participants agreed that the problem was irrelevant. If no commitment or agreement on condition was reached in a situation the designers were prepared to intervene and try to reinterpret the situation to reach consensus. Such an intervention did, however, not occur during this game.

By the end of the first day the really hard work for the designers start. As preparation for the second day we update and print a new version of the playground and all the role scripts with all the commitments and conditions that have been agreed upon during the day.

The second day starts with a review and a discussion of the updated playground and the new role scripts. The rest of the day the group work with the remaining situation cards in the stack. To get some variation and more involved participation the group split up into smaller groups in between.

By the end of the day *Act II* is prepared. The group is now asked to select three different publications that are typical for their present production. At Konsumentverket the group choose a catalogue with a complicated layout, a book, and a newsletter. As preparation for the next act both each participant and the designers write five new situation cards related to their experience of working with these publications.

5.4 Act II (2 x 8 hours)

A new gathering on the stage. The scene has changed. Now the play comes closer to the situated work at Konsumentverket. The new publication related cards reminds the participants of earlier breakdown situations. However the task in the game is now to "replay" the chosen publications in the light of roles, commitments and conditions agreed upon in Act I.

During the first day the playground is adjusted for each of the publications and the new relevant situation cards are played. At Konsumentverket this turned out to be much more complicated for the participants than the work in Act I. For example one of the new cards on the publication "the catalogue" read: "The catalogue was scheduled for the annual education fair, but due to production problems the catalogue was delivered to late to be printed in time for presentation at the fair." This situation card led to an animated discussion on who´s responsibility it was, rather than commitments on how to solve the problem given the new roles.

Given this breakdown the designers decided to introduce a change of the rules of the game as they played along. Hence, on the second day a new type of cards labelled "development cards" was introduced by the designers. The development cards were formulated as a response to problems that arose during the "replay" of the publications. For example the development card "How can we improve proof reading of our publications?" was generated as a response to a qualification, and a task, that seemed to be underestimated in the organization. The discussion of the card led to an acknowledgement of this problem, a person that committed her to solve it, and an agreement from the group that this was an essential task.

5.5 Act III (2 x 8 hours)

By the end of Act II about 150 commitments and conditions have been formulated and agreed upon. As preparation for Act III, the playground was updated with these agreements by the designers. In Act III the players prepare to meet their audience, and earlier commitments and conditions are tied together in an *action plan* for negotiations with the surrounding organisation.

The designers introduced six headlines for refining the commitments and the conditions into demands in the action plan. The demands concerned:

• *publication program* - how to develop the products,

• *professional support* - ideas and proposals for competence which must be, "imported" into the organization, e.g. typographical competence,

• *development of qualifications* within professional roles,

• *technology support* on assessment, investments, and maintenance of the technology,

• *organizational support* concerning relations to the surrounding organization and management,

• *internal work practices* - new routines, quality control stations, and team work the participants can implement themselves.

This structure for the action plan was based on the designers earlier experiences with making action plans with users. A certain family resemblance with the elements in West Churchman's *systems approach* is also evident [Churchman 1968]. The structure was a suggestion to the participants, a suggestion that could be changed and modified. At Konsumentverket, however, the participants accepted the six headlines.

For every single work situation on the playground commitments and conditions are cut out and pasted into the action plan under the demand headline the group finds appropriate. For example in the work situation "formatting" the technical support person had committed her to continuously update all desktop publishing software if she was offered appropriate training and time to do the job. This commitment was pasted under the demand headline "technology support" and "development of qualifications".

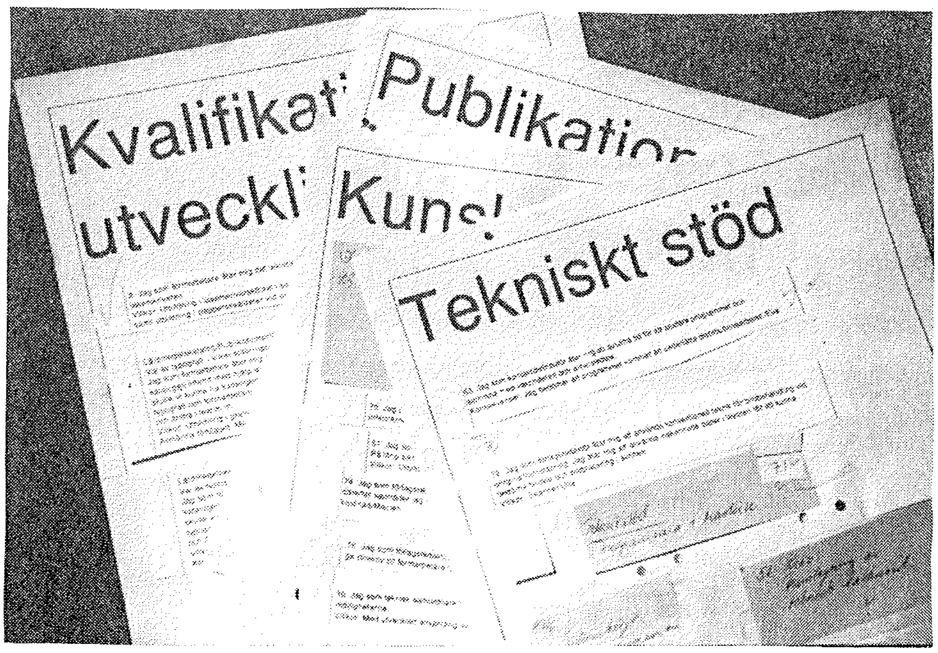


Figure 6. The action plan at Konsumentverket in progress. Formulated commitments and conditions from the play are refined under titles like "development of publications" and "development of qualifications".

The last day was dedicated to scrutinizing and refining the demands in the action plan. Every single commitment and condition under each demand headline was evaluated, given a priority, and debated to find good arguments. Only demands on which group consensus could be reached were kept in the action program. For the five first demand headlines these demands were the groups *requirement specifications* to Konsumentverket for changes. For example, the group demanded that the technical support person should get proper training on desktop publishing software, and software support as part of her job description.

The last demand headline "internal work practices" was treated differently. These were the demands the group could realize without external support. Not surprisingly a new playground with concrete guide-lines to support the new work practices was made by the participants. The playground was transformed as a script for action in reality. The designers were now made redundant. At least in that sense the game was a success.

6 Review of the Game

Looking at the game in retrospective these are some of our detached reflections:

6.1 The design-by-playing approach

Did the design work at Konsumentverket benefit from the design-as-playing approach?

No doubt there was a *playful engagement* from the participants. Most participants were very engaged and active through the process. The normative point of departure was well supported by the game.

The methodological *game and play metaphor* was appreciated by the participants, but playing-in-reality often meant that reality rather than play set the conditions for the design work, the design work oscillated between play and reality. The play situation created a distance to the everyday situations. At the same time it had a strong family resemblance. The rules of the game were all the time at stake and changed during the game. Was this a failure or the way these games should be played?

For us as designers *the language-game philosophy* has been fundamental in creating the Desktop Publishing Game, and in reflecting about what it meant. The design game was created as a game that had family resemblance with the ordinary work of the users and with their experience with games and plays as well as with our practice as designers. Reflecting about the design process in terms of e.g language-games, speech acts, linguistic artifacts and reminders made sense to us as designers.

6.2 Words of critique

The design-by-playing approach we have used is very *subjective*. This is both a strength and a problem. The strength is that both designers and users become visible and responsible in the design process. No one can hide behind a formal description.

On the other hand this subjectivity makes the approach very work intensive for the designers, especially in the preparation work between the acts since the quality of the play is totally dependant on these preparations.

To support the designers in this preparation work some computer support for making and updating playgrounds and role scripts was developed. The idea was actually to support on the spot updates during the meetings. The developed computer support was, however, to slow and complicated for this.

The approach is also subjective in the sense that the outcome is very dependant on the personalities of the participating individuals, and their social relations, not only their professional ones. In fact, one of the problems for the designers is to distinguish between the two, and guide the game towards professional commitments.

If we look at the methodological aspects of the design-by-play approach we can conclude that the generality is very low. Every new case is not only a new application, but the form and content of the game as such must be reconsidered. For the designer this means a severe difficulty. There is no safe methodological ground to stand on, experiences from earlier cases and sensibility for the special aspects of a new situation are decisive.

Another aspect of the low degree of generality is the question of how the experiences of the game can be defused in the organization. The design game is most engaging for the participants, the new insights are "embedded" in the group. Diffusion does not occur through written documentation and oral presentation. It seems that "replaying" the game with new participants is a necessary condition. But at these replays the professional designers have long ago left the scene and lay designers carry on the new games.

The design-by-playing approach is time consuming. Hence, in words of critique it is valid to ask if the game is worth its price. We estimate the time spent by us as professional designers to 30 days including all preparations. The participants from Konsumentverket spent all together approximately 60 days with the game. In costs this can be estimated to about one-third of the investments at Konsumentverket in desktop publishing technology. In addition there will be substantial costs to conduct the suggested education program. We are not able to say if the price is to high, but compared with real organisational and educational costs in traditional systems development the cost does not sound alarming to us. In any case, without the investments in education and organisational design via the desktop publishing game at Konsumentverket, it would probably have been a more economic idea to forget about the investments technology as well.

6.3 Konsumentverket today and other applications

After the design game ended in October 1988 we have had a follow-up meeting in December 1988, and discussions with individual participants during the spring of 1989. According to the participants they have sorted out many of the responsibility aspects. They have managed to structure the publication process in a more sufficient way than before. They have established new professional routines. They make fewer mistakes and have a higher ability to organize the publication work. According to the print shop they cooperate with, they have improved their competence as purchasers of printing services and raised the quality. They also still use the playground, but now as a tool to structure the planing work and to get an overview of the actual production situation. The other side of the coin is that they, despite this progress, not have been able so far to fully utilize the potential functionalities of the new desktop publishing technology.

As for the designers there have been new design games and practical situations to participate in. In one case a new Desktop Publishing Game has been played. A game with family resemblance with the old one, but certainly a new and different situation, requiring new rules of the game.

However, we have also entered new application domains, domains in which we have very little competence as opposed to publication work. One of these games concern the development of organizational alternatives in a police precinct. In another case the game approach with playground and situation cards is used as a general methodology to improve professional roles and relations in the organization and to the customers. As designers we instruct participants from the organization in the design-by-play approach, and they are the ones that initiate and guide the actual design games in the organization. To the results of these new games we will have to return in another paper. No doubt, there will be new surprises as we play along.

6.3 Redesigning the game

In a workshop at a conference on "creativity and systems development" the ideas of the game were presented to researchers in the field. The workshop had as its theme the editorial work of the *Scandinavian Journal of Information Systems*. Among the many suggestions for improving the game we found the following ones particularly interesting.

We were criticised for the rigid structure imposed by the playground. As a remedy it was discussed to extend the prologue by active user participation in the construction of the playground.

One way to do this could be to use the techniques for making "rich pictures" in Checkland's soft systems methodology. As designers we could support the participants themselves to draw a "rich picture" playground identifying slow-to-change structures and continuously-changing processes and how these interplay and form the climate of the situation [Checkland 1981].

Another way to improve the social construction of the playground could be to use elements from the Future Workshop methodology, especially the techniques used in the critique phase and the formulation of themes for positive alternatives [Jungk and Müllert 1981].

We are actively going to try out possibilities of transforming such redesign work of the game into reality. But we would be more than happy if others socially construct new design games that have a family resemblance with the ones that we have played.

Postscript on the Game of Scruples

Finally, we would like to present the reader with a situation card: "Who will win the Desktop Publishing Game, the manager or the secretary?". This is certainly an ethical dilemma that gives us scruples. Is design-by-playing a game of seduction rather than one of liberation?

References

[Ackoff 1974] Ackoff, R.L.: *Redesigning the Future*, John Wiley, 1974.

[Churchman 1968] Churchman, C.W.: *The Systems Approach*, Delta, New York. 1968.

[Checkland 1981] Checkland, P.B.: *Systems Thinking, Systems Practice*, John Wiley, Chichester 1981.

[Ehn and Sjögren 1986] Ehn, P and Sjögren, D.: "Typographers and Carpenters as Designers" in *Proceedings of Skill-Based Automation*, Karlsruhe 1986.

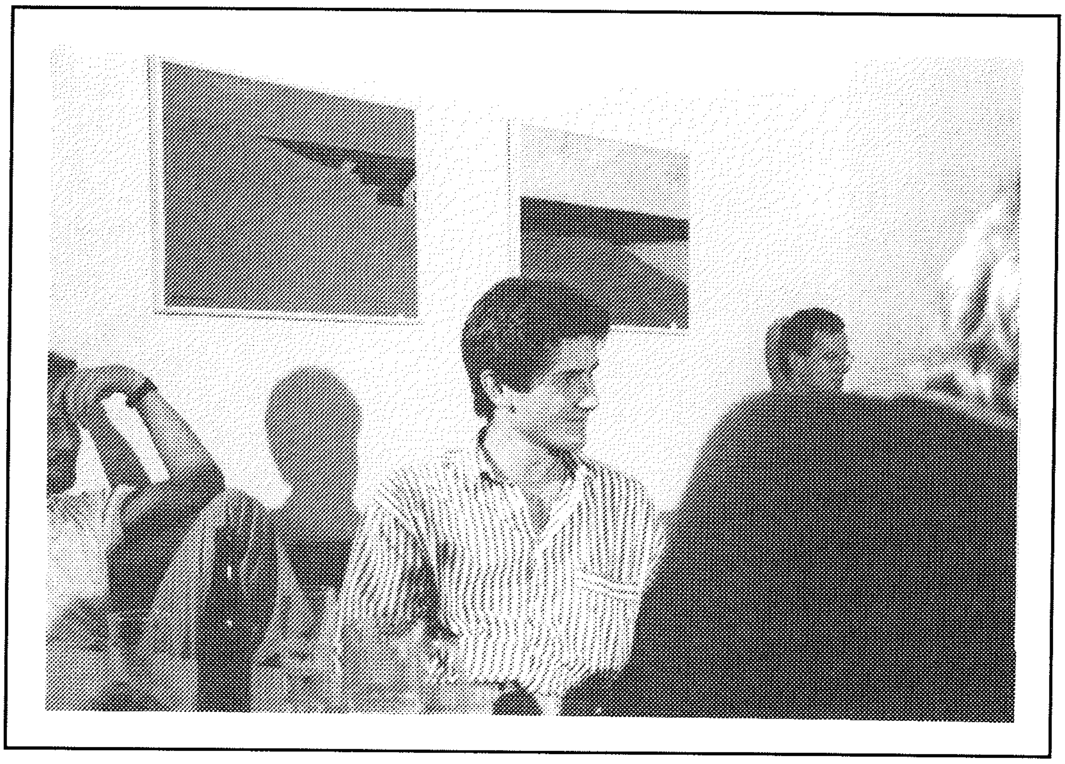
lEhn 1988] Ehn, P.: *Work-Oriented Design of Computer Artifacts*, Almqvist & Wiksell International, Falköping 1988.

[Jungk and Müllert 1981] Jungk, R and Müllert, N.: *Zukunftwerksttäten - Wege zurt Wiederbelebung der Demokratie*, Hoffman und Campe, Hamburg 1981.

[Winograd and Flores 1986] Winograd, T. and Flores, F.: *Understanding Computers and Cognition – a new foundation for design*, Ablex, Norwood 1986.

[Wittgenstein 1953] Wittgenstein, L.: *Philosophical Investigations*, Basil Blackwell, Oxford (1953) 1963.

[Wynn 1979] Wynn, E.: *Office Conversations as an Information Medium*, University of California, Berkeley 1979.



Preben Mogensen, Mads Nørby, Kim Halskov Madsen

A Visual Simulation Model of Information and Work Systems

Inger V. Eriksson[*]
Annika E. Finnäs[o]
Åbo Akademi University
Department of Computer Science
DataCity, SF-20520 Turku, Finland

Abstract: To facilitate the use situation an interface that supports description of the information system, the structure of the organization, and division of labour is proposed. A Help-system with a visual simulation model is an important part of this interface. The simulation model allows the user to follow the transactions on different hierachical levels of detail; transaction flows between departments and units, detailed manipulation on the section level, and step-by-step progress of computerized functions. The time dimension is taken into consideration and the transactions can be followed forward but also be traced backward. The backward trace is accomplished by two history files concerning data and actions taken. The model is designed to be used in two modes: simulation controlled by the system and simulation controlled by the user, controlled and interactive simulation, respectively. A prototype version of controlled simulation is presented as an example. It is implemented on Macintosh using Hypercard.

1. Introduction

Graphical representation such as visualization and animation, and intelligent Help-systems are modern concepts. Improved techniques give the opportunities but the motives for use are more serious. Many organizations are strongly dependent on their information systems to run their business. However, it is well-known that there are problems in the use situation. These problems partly emanate from inadequate knowledge of the computer-supported work. Users might have received training in practical use of the computer to perform certain specific tasks but do not understand the overall context. This lack of overview as well as the integrated

[*] Inger Eriksson is a researcher in the SOLE project (SOftware Library Evolution) at the Åbo Akademi University and a member of the Knowledge and Work research group.
[o] Annika Finnäs is a research assistant in the Xtend project at the University of Turku and a member of the Knowledge and Work research group.

structure of many information systems conceal the organizational interaction between work and information systems. Division of labour and collaborative conditions are controlled by information systems. However, this role of the information systems does not seem to be sufficiently clear to the workers. Moreover, most of the description techniques of information systems have the weakness that they concentrate on the information system (often even the computerized system solely), neglecting the work context in which the system is to be used.

One additional explanation for the difficulties might be differences in the cognitive models of the designers and the users as regards the system in question. The source of the problems can be traced back to the design process. System designers describe the system from their point of view, which is often rather technical and formal. The users, although they might define the requirements and accept the system, do not necessarily interpret it in the same way as the designers do. Their version of the system is based on the concrete use situation with all its exceptions and special cases, which are not easily formalized. There thus exists a conflict between the designed system and its interpretation in the use situation.

Our ambition with this research is to improve overall understanding. Another important goal is to eliminate or diminish the gap partly between interpretation and reality and partly between the users' and designers' models. For this purpose we offer a complex model of reality, including both the work and the information system. We also make this model concrete by using simulation and visualization, not just verbal descriptions. Such a context-sensitive and perceptible simulation model supports acquisition of a total perspective. A prototype version of a Help-facility for an inventory management system application is designed to test the value of our ideas. This example is presented in this paper to illustrate the discussion.

The remaining part of this paper is organized as follows: In section 2 we give a background for our approach. In section 3 the theoretical basis of the visual simulation model is discussed. An example is presented in section 4. Further development of the model is discussed in section 5, where future prospects are also examined. Section 6 sums up our experiences and expectations.

2. Background

In the Knowledge and Work project[1] we wanted to improve the use situation by increasing the users' understanding of the information system [Eriksson et al., 1988]. This was done by supporting learning and one of the techniques used was simulation; role-play and manual

[1] The Knowledge and Work project (1985-1989) is the first project of the Knowledge and Work research group at the University of Turku and Åbo Akademi University.

simulation of the computerized functions. A Help-system designed to support the whole work situation might be another way of mediating this kind of knowledge, both deep and general.

In the Xtend project[2] the idea is to create an interface with facilities that support description of the information system, the structure of the organization, and the division of labour. It aims to give an overview of the organization and promote users' general understanding of how the system functions as well as improve their knowledge and skills for collaboration. For this purpose up-to-date information supportive for performing the ordinary tasks, presented in an adequate way, must be offered.

We propose a complex interface including a Help-system, an important part of which is a simulation model. The idea is that the user can simulate different transactions. Changes in data are shown as well as how they are reproduced. The simulation facility makes it possible to follow the transactions forward but also to trace them backward. The user can also put what-if questions (or rather make experiments) about the future and how-did-we-arrive-at-this-position questions about the past.

The simulation is performed in two modes: simulation controlled by the system, controlled simulation, and simulation controlled by the user, interactive simulation. The user can also choose to follow transactions on different hierarchical levels of detail. The base is a map, that is, a structured description of the organization with the departments and units concerned, where the transaction flows between these are shown. This kind of map structure was used in the Florence project [Bjerknes et al., 1987] and evaluated as the most natural way of representing the organization to the users. If the user wants she[3] may choose to look at the transaction on the detailed level of each section. For deeper understanding of the computerized functions it is further possible to see how these operations are performed step by step.

Controlled simulation is primarily designed to be used in situations where new material is to be learned. This may mean that a new system is implemented or new workers are employed. Interactive simulation, on the other hand, is designed to support every-day work situations; to assist in fault-finding and error recovery, and to master unusual situations. Simulation and tracing the transactions backward as well as access to data of the history of each object (from its creation till it leaves the system) are helpful for this purpose. The Help-system is thus designed to offer a means to update old knowledge as well as to acquire new.

---

[2] The Xtend project (1988-1990) is a continuation of the Knowledge and Work project, led by Professor Markku I. Nurminen, and mainly financed by the Finnish Work Environment Fund, and also by the Doctoral Education Programmes of Information Technology in Finland.

[3] For practical reasons we consistently use *she* and *her* instead of he/she, his/her, him/her etc in this paper but naturally 'she' alludes to 'he' as well if the context does not imply anything else.

To gain some experience in design of this kind of interfaces and to test the usefulness of the vision an example case had to be selected. One of the cases studied in the Knowledge and Work project, an inventory management system, was regarded as suitable for this purpose. Based on this system a prototype was designed.

3. Theoretical Basis

Our frame of reference originates from the central ideas of the Knowledge and Work project [Nurminen et al., 1987]: to uncover the social interpretation of information systems, to make the users of the systems visible and to show their responsibility as actors. We wish to eliminate the borderline between primary work and knowledge work, also its computer-supported parts, and show that these are too intertwined to be treated as separate matters. This is the ideological basis while the more functional ones, user interfaces, Help-facilities and visualization, are discussed in the following.

User Interfaces and Help-facilities

There are at least two metaphors that describe ways in which human interact with computers: the conversational world and the model world. In the conversational world some command language is used to describe what to do. This type of dialogue is called sequential dialogue. In the model world a user shows what to do by manipulating visual representations of objects. Shneiderman [Shneiderman, 1988] uses the concept 'direct manipulation' for this interaction style. The corresponding dialogue is called asyncronous dialogue. Associated with direct manipulation is also a task-oriented concept, multi-thread dialogue. And further, the concept concurrent dialogue refers to multi-thread dialogue where simultaneous execution of several tasks is possible. Only the most relevant concepts from the point of view of our research, discussed by [Hartson and Hix, 1989] are mentioned here.

Two opposite approaches how to design user interfaces are also present: integration versus separation of the design of the dialogue and the computational systems. As to the integration approach the discussion concerns integration of human factors but the main interest is exactly on user interfaces. Changes to both traditional and prototyping lifecycle design approaches are proposed [Mantey and Teorey, 1988; Summersgill and Browne, 1989]. On the other hand, reasons to separate the design of the interface and the application parts are also presented [Hartson and Hix, 1989]. They also mention systems for which separation of dialogue from computation was attempted after the system was implemented. That means that user interfaces were developed to be used as 'add-on' front-ends to existing application systems.

An important part of the user interface is the Help-facility. The traditional one is some kind of index which describes the used commands. However, also guided tours have already "traditions". Trigg [Trigg, 1988] refers to Vannevar Bush's article from 1945[4] where the notion of a "trail" is described. Hammond and Allison[5] also describe a tour-like capacity but they further distinguish between two modes, learner controlled and system controlled. By guided tour they mean the system controlled mode. Zellweger[6] talks about active paths which connect objects in a multimedia document. On each stop along the path the system can perform actions. Weier and Borning[7] have implemented active tours but they also include support for interactive simulation in their electronic encyclopedia. NoteCards, a hypermedia system developed at Xerox PARC, uses guided tours to support communication between the author and the reader of a hyper document. Most of these guided tour facilities are restricted to static presentation. An interesting and challenging feature is to give users control over certain aspects of the dynamics of the tours. One example of work in this area is Brown's[8] BALSA-II animation system. Through its "programming by example" interface a user can dynamically effect changes in programs although she had little or no knowledge of the programming language.

Visualization

Visualization is generally a powerful tool to give an overall view. Its usefulness, almost necessity, in describing parallel programs is widely accepted. The real world certainly functions in a parallel way so simulation/animation could also be used beneficially to describe it.

There are three dimensions of visualization: visual programming, program visualization and data visualization. The latter two are of interest in this context and are briefly described in the following. The purpose of program visualization is to offer a clear and correct understanding of the structure and function of a program [Brown et al., 1985]. Visualization may be static or dynamic. Static pictures are used to describe the state of the program at a given point or factors which are unchanged over a prolonged period. Dynamics are used to describe how execution of the program proceeds. The object to be visualized is a computer program which is actually an algorithm on a detailed level. However, an information or work system may also be described by algorithms although on a higher hierarchical and more abstract level. Of course, exceptional

[4] Bush, V. (1945). As we may think, in The Atlantic Monthly 176, 1, June 1945, pp. 101-108.
[5] Hammond, N. and Allinson, L. (1988). Travels around a Learning Support Environment: Rambling, Orienteering or Touring?, in Proceedings of the ACM CHI '88 Conference, ACM, New York, pp. 269-274.
[6] Zellweger, P. (1988). Active paths through multimedia documents., in J. C. van Vliet (ed.), Document Manipulation and Typography, Cambridge University Press, New York, pp. 19-34.
[7] Weyer, S. A. and Borning, A. H. (1985). A prototype electronic encyclopedia, in ACM Transactions on Office Information Systems, Vol. 3, No. 1, pp. 63-88.
[8] Brown, M. H. (1988). Algorithm Animation. MITPress, Cambridge, Massachusetts.

and unforeseen situations which are integral parts of these systems cannot be treated in a formal way. In an algorithm such events might be handled, for example, by statements such as: "ELSE PERFORM EXCEPTION HANDLING". With this restriction the discussion above suits our purpose.

Data visualization means visual description of databases. Graphical representation is used to adapt the interface to different categories of users who may have individual needs and various background knowledge [Larson, 1986]. This aspect is exactly what we regard as necessary in ordinary use situations. Depending on the transaction, different types of information are needed. Performing the tasks requires detailed information. To be able to cope with, control, accept, and forward the transactions it is necessary to have some knowledge of the work and information needs of other interdependent sections. The depth of knowledge as well as its breadth thus vary with the situation. Further, information ought to be presented in an adaptable and useful way since there are users on different levels of the organization who need different aspects of the same information for their work. Some data should be accessible and subject to manipulation by a specific group. Other data may be seen but not manipulated and, further, there are data not accessible to them. For efficient use easy-to-operate functions are necessary and here graphical representation may be beneficial.

Benefits of Simulation

It was already mentioned that problems in a use situation are common and the users are expected to cope with these. Human ability to adapt to peculiarities in system performance and to optimize interaction is the very reason for having people in system and not to strive for full automation. To take full advantage of these human abilities it is important to have opportunities to "cut corners" and to perform trial and error experiments [Rasmussen, 1987, p. 25]. Simulation included in a Help-function offers a safe way of experimenting for users.

Another great advantage of simulation is that concrete pictures of abstract concepts together with their relations can be created. Also abstract models of the factual reality can be described in a very concrete way. This improves the real understanding of phenomena and tasks. Learning the underlying processes on an abstract level is much more valuable than learning to treat each specific case in a given manner. Understanding principles gives knowledge on a deeper level and makes it possible to derive the correct way of performing unusual tasks [Eriksson and Reijonen, 1989]. As regards the use of information systems this means that users can cope with exceptional and specific situations, not just routine ones. Our idea of a visual simulation model of information and work system offers such new aspects which improve precisely this kind of knowledge acquisition: It offers a system transparent enough to show what really happens within it, thus facilitating understanding of the basic concepts.

4. Description of the Prototype

Our aims placed certain specific demands on the programming environment. The object-oriented approach had features suitable for describing the model. To make the offered Help-facilities as informative and easy to understand as possible graphical representation was considered necessary. Since we wanted to simulate transactions in time, animation seemed to be a natural way of doing this. Our demands on the environment were thus object-oriented design/programming with good graphical and animation properties. We decided to use the HyperCard™ tool supplied on the Macintosh.

The System

In this paper we use an inventory system in an organization in the food industry to illustrate our ideas. Since the work in the inventory is dependent on other departments, we include some of these secondary departments, too, namely the order processing department, the manufacturing departments and the packing unit. Also certain links, eg., with sub-contractors and customers, are of interest.

The inventory department is divided into two main sectors: the bulk inventory and the buffer inventory. The bulk inventory serves as a long-term stock and the buffer inventory as a short-term one. Several functions in the bulk inventory, for example, the FIFO-order of stored goods and the organization of shelf positions, are computer-supported [Eriksson et al., 1988].

Our model is based on this system but we have made certain assumptions and improvements in the real system in order to emphasize clear boundaries of responsibility and information ownership. Whole work tasks, including both primary and knowledge work, are also kept distinct. This reconstruction corresponds better with the ideas of the Knowledge and Work project [Hellman, 1987].

Figure 1 shows the structure of the reconstructed system. There are three parallel manufacturing/packing (M/P) units, each of which has its own quarantine sector. The product pallets are transported from the M/P units straight to the bulk inventory, or via the quarantine. In our prototype the bulk inventory has a shelf capacity of 28 shelves for storing the pallets, which is only a fraction of what the real system has. The shelves are divided into 5 categories according to their location, size, strength etc. and a product has up to three shelf categories associated with it. This determines which products can be placed at each location. Further, there are a few shelf positions that do not belong to any specific category and therefore can be used for storing any product. If the shelf space is not sufficient, however, we have assumed that there is always space enough along the walls.

The pallets are transported to the buffer inventory from the bulk inventory or directly from an M/P on request. The pallets are stored in the buffer inventory at fixed shelf positions according to the product number. Products from sub-contractors are delivered either to the buffer or the bulk inventory.

Figure 1 also shows all possible routes a product pallet may take within the inventory and which associated virtual storages are concerned. The virtual storages are logical storages and represent one solution to keep the quantity in stock up-to-date and the domains of responsibility accurate.

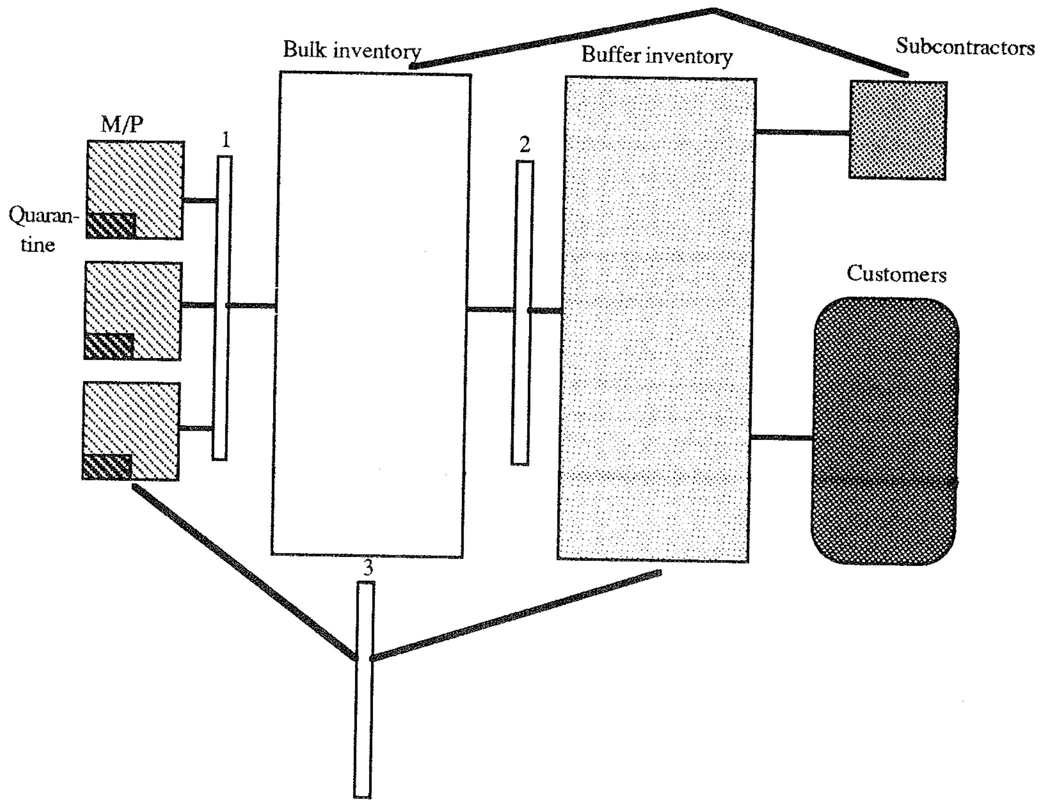


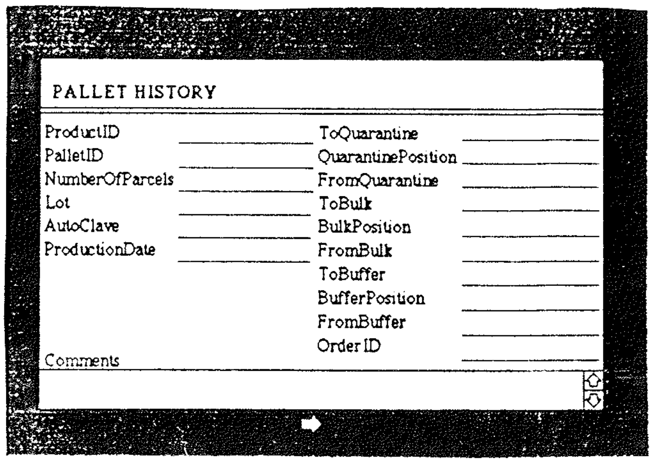
Figure 1. The structure of the inventory system.
1. Virtual storage between quarantine and bulk inventory.
2. Virtual storage between bulk and buffer inventory.
3. Virtual storage between quarantine and buffer inventory.

The Model

Our purpose is to describe the actions (manual and computerized) in the inventory in a surveyable and easily understood way. This is done by creating a simulation model of the system and by representing the model visually. The design of the user interface is kept distinct from the application and relies upon the idea of direct manipulation (see section 3). Once the system is started up there is no need to use the keyboard. All interaction is done with the help of the mouse. Advice is offered through guided tours where actions and changes in data are represented visually. In addition to the animation the user is also given written information about what is happening on the screen and what action she is supposed to take from a text field at the bottom of the screen. We could think of three different ways of representing the model on the screen. One is to divide the screen into two windows and show the material flow in one window and the information flow in the other. This would give a well-structured description but the size of the screen made us reject this idea. Another method is to use the whole screen to show the material flow and then switch to another and show the information flow. This method would, however, make it difficult to obtain an overall understanding of the whole work context. The third method, which we finally decided on, is to mix the material and information flows on one screen. This approach supports the principles in not separating primary work and knowledge work and also gives an overview of the activity.

The model is described on three levels of detail. On the first level (overview level), the transaction flows between sections of the inventory system are shown. On the second level (section level), the manipulation of a transaction within one section, for example the bulk inventory, is represented. On the third level (computer level), the user is shown how the computerized functions are performed in a detailed way. At this level the user gets the chance to follow exactly how the activity would be performed if done manually. We do not describe the actions in this way because we think that it would be easier or even possible to perform them manually. However, we think that using well-known concepts, such as manual files and cards holding information makes the description easy to understand. We also think that this kind of description can bridge the gap between the formal way of treating data by the computer and the user's more flexible interpretation of information.

The model is also described in the time dimension, which means that it is possible to follow a transaction forward and backward. The backward trace is realized by two different history files. The first one holds information about the route a product pallet takes through the system, starting from the moment the product is produced until it is finally delivered to a customer (Figure 2). The *comment* field can, for example, be used if a product pallet is returned by the customer. The other history file consists of central pictures stored during the simulation of the transaction.


PALLET HISTORY

ProductID		ToQuarantine	
PalletID		QuarantinePosition	
NumberOfParcels		FromQuarantine	
Lot		ToBulk	
AutoClave		BulkPosition	
ProductionDate		FromBulk	
		ToBuffer	
		BufferPosition	
		FromBuffer	
		Order ID	

Comments

Figure 2. The Pallet History card.

An Example

Here we describe one part of the simulation model: controlled simulation of one transaction at a time. Two different types of transactions in the bulk inventory are described. First we give a description of the actions taken when a product pallet is delivered from an M/P unit to the bulk inventory. This is followed by a survey of the actions associated with an order from the buffer inventory. Appendix A gives a detailed description of the former transaction.

The following actions take place when a product pallet is delivered to the bulk inventory from an M/P unit:

- The amount in quarantine is updated (if from quarantine).
- The amount in the virtual storage between quarantine and bulk is updated.
- The pallet is brought to the bulk inventory.
- The pallet is acknowledged.
- The amount in the virtual storage between quarantine and bulk is updated.
- The amount in bulk is updated.
- A suitable shelf address is chosen from the category data.
- The shelf position information is updated.
- The pallet is brought to the given shelf by a fork-lift truck.

The simulation is started from the overview level where the user is shown how an arbitrary product pallet is delivered from either of the three M/P units. At this time the object *product pallet,* that is to say, the history card for the pallet in question, is created (Figure 2). The quantity in stock in the particular quarantine section is reduced and the virtual storage between quarantine and the bulk inventory is increased. These actions are visualized by showing the user how the appropriate fields in the corresponding files are updated. After this the user can choose whether she wants to see the detailed manipulation of the transaction at the section level or if she wants to continue at the overview level. Let us assume that she wants to see what happens to the pallet in the bulk inventory. There the product is acknowledged, the virtual storage between quarantine and bulk, and the storage in bulk is updated, and a suitable shelf position is searched for. The algorithm for this follows:

- •Look in the product information for suitable categories for the product in question.
- •For each of the categories from the previous step
    - -look for shelf positions belonging to the appropriate category in the category information,
    - -look for free shelf positions among the positions from previous step in the position information.
- •If no free position is found, look for a free position among the shelf positions that do not belong to any specific category.
- •Choose a free position, or if none is found, choose the "wall" position.

As the search continues the user is first shown all suitable shelf positions and then one of these is automatically chosen. The shelf position information file is updated and so is the history card for the product pallet in question. After this a shelf position label is printed and stuck on the pallet which is transported to the chosen shelf position. According to the updating of the storage in the bulk and the search for suitable shelf positions the user has the ability to move one level deeper and see how these actions are performed in detail.

The other type of transaction to be dealt with is the reception of an order from the buffer inventory. The following actions take place in the bulk inventory when they receive an order:

- •Look for appropriate product in the bulk position information.
- •If none is found, a message is sent to the buffer inventory and control is returned to the overview level.
- •The product pallet is taken out of the shelf position and the shelf position information is updated.
- •The amount in bulk is updated.
- •The amount in the virtual storage between bulk and buffer is updated.
- •The product pallet is delivered to the buffer inventory.

Starting this transaction causes an arbitrary product chosen from the product information to be ordered from the bulk inventory. The product ID of the ordered product is shown to the user and the order moves inside the bulk inventory. A search is made through the bulk position information to find out whether there are any of the product asked for. If none is found, the buffer inventory is informed and the simulation terminates. If the product is found, the user can move to the section level if she wants to see how this transaction is manipulated in the bulk inventory. There the user is given the pallet ID's of each pallet, of the product in question, in the bulk inventory. However, the user cannot in this case (controlled simulation) choose which pallet is to be delivered to the buffer inventory. Instead, the FIFO-order is strictly followed. That is, the pallet that has been in the bulk inventory the longest time is automatically chosen. When the pallet is chosen for delivery, the shelf position that holds the pallet is highlighted. The shelf position information, the history card for the product pallet in question and the amount in stock in the bulk inventory is updated. After this the scene returns to the overview level where the amount in stock in the virtual storage between bulk and buffer inventory is updated. Now the user is shown how the pallet ordered is moved outside the buffer inventory.

Experiences of the Process

HyperCard appeared to be a flexible tool for animation. Among the methods and combinations of methods for creating animation, one that suits the needs can almost always be found. The size of the screen, however, restricts the possibilities. Since the commands of the scripts are interpreted one by one, program development and test process are fast but the speed of execution is limited. In particular when moving between different stacks, execution often becomes unacceptably slow.

Documentation of Hypercard applications seems to be a general problem. How can a good and readable documentation of scripts on different objects in HyperCard be created? As far as we know, there are no standard or even recommended methods in use. The idea of sharing information becomes important when several persons work together on a prototype. This is difficult, however, in HyperCard since the information is spread over different levels and even over different objects on these levels. In the project work, however, these problems must be overcome in some way.

5. Future Prospects

In the previous section controlled simulation of one transaction at a time, was presented. The second mode is interactive simulation of the same type of transaction. Here the user herself can test different ways of performing the tasks. Consequences of the choices are given. Impossible/prohibited actions are not allowed, nor is the wrong order of performing them. However, an explanation why these moves cannot be done is offered.

The second level of simulation, in terms of sophistication, deals with controlled versus interactive concurrent transactions. Here priority aspects must be addressed. This level has not yet been implemented. The third level is meant to handle situations where several actors perform their tasks at the same time on the screen. This is, however, difficult to implement in the technical environment we have chosen to use. For this phase more powerful equipment would be needed.

The prototype discussed in this article is at present still under construction. We plan to have it in test mode until autumn '89 when it will be evaluated and compared with a more traditional work environment. In this case we cannot have the ordinary users to test the prototype since the system has been modified as mentioned in section 4. The evaluation concerns learning, use, understanding, and motivation. We expect simulation to facilitate learning as regards the time needed and the number of errors made. As to the use situation we are interested in finding out if users experience simulation helpful and if tasks are performed more quickly and with fewer erroneous actions. Changes in users' understanding of the function of the information system will be studied. Also their interpretation of their own position, tasks, and domains of responsibility within the work system, is of interest. If learning to use the system, by the help of simulation, becomes easier, we expect the motivation to use the system to improve, the resistance to new technology to diminish, and also the individual's evaluation of their own work to be enhanced.

However, it is generally accepted that new solutions create new problems. It is thus justified to ask which and where the risks are with this facility. We are not changing the system itself; it will continue to work exactly as earlier. Instead, we offer a new opportunity which may be used or not, depending on the user's choice. The simulation model is a supportive function but not necessary for task performance. Thus, the only risk with the new situation confronting the users is, as we see it, that the gap between the proficiency of individuals might increase. There might be those who can make use of the Help-system and master their tasks well or even better than without the assistance of the system. There might also be those who do not feel comfortable with a computer and who do not want to use it more than absolutely necessary.

These persons find themselves in a difficult situation: they know that there is help available in problematic situations but cannot take advantage of it. Thus also their primary work can suffer compared with that of others. However, we offer an easy-to-use interface to diminish antipathy to computers and promote the use of the system. We have also made an effort not to introduce any unnecessary new concepts, but use those of the workers and those already applied when using the ordinary system. These actions, we hope, makes use of the supportive system easy and attractive enough not to hinder anyone from taking advantage of it.

Generalization

In this paper a specific case was presented as an example. However, generalizations of both practical and theoretical interest are possible. On the one hand, this new type of user interface could be further developed, in the light of the evaluation of its usefulness, and result in guidelines for designing interfaces to support the work tasks as a whole. Certainly a practical implication with general consequences.

On the other hand, it is possible to abstract certain central concepts from the case presented. *General transaction objects* with their internal properties, eg., their initial and final states, and a *network of transactions* could be specified. A *coupling matrix* describing the flow of the transaction objects through the network forms the control structure of the system. The actual system might then be generalized by parametrizing its properties. By such system-specific parameters tasks can be defined as preceding, current and following or as sources (which start the transaction), intermediates (which are trigged further step by step) and sinks (which terminate the transaction).

A general simulation model could then be described in the following way: some kind of a trigger (impulse or event) is needed to initiate a transaction which might be the handling of material or information objects. This trigger is defined from two points of view: there is the logical aspect, where given conditions and states of the object are enlisted, and the task aspect, which signals that the next task can be started if the logical conditions are fulfilled and all the preceding tasks are performed. The first organizational section in the chain acknowledges receipt of the transaction and takes over responsibility for it, performs some "manual" and/or information tasks, and when finished, delivers it to the next section in the chain. This, in its turn, acknowledges the transaction, takes over responsibility, and so forth, until the object leaves the system. Thus both the material object (if any) and the corresponding information are accepted, treated, and delivered further in parallel. Naturally this description of a work process is very schematic but that is the price of streamlining and generalization. However, realization of general simulation models as hypothesized above requires another and more powerful environment than the one used in this case.

6. Conclusion

We have been discussing visual simulation and its benefits in supporting the use of information systems in work context. A simulation model as a part of a Help-system has been presented. We have also discussed possibilities for further development and generalization based on this case.

The efforts described are derived from the basic ideas of the Knowledge and Work research group: information systems should be designed in such a way that they make the actors visible and improve mastering of the work situation as a whole. Systems of today are seldom designed to meet these demands and that is why a Help-system might be supportive in the use situation. The model discussed in this paper certainly supports the idea of real human actors using information technology to assist them in their ordinary work tasks. The model provides a perspective which covers the users and their work within a wider environment. It shows the boundaries of each user's responsibilities but also gives an insight into the domains of other sections and how these are all interconnected, in other words an overview.

This type of broad understanding can be expected to improve the quality of the work situation. It is easier for the workers to see the consequences of their actions and how they themselves are dependent on others. Cooperation is thus facilitated and stimulated. Also fewer errors and more consideration towards others might be a positive effect. This affects the quality of the production results. On the subjective level qualitative improvements are to be expected, too. If the user understands her position, importance and responsibility in the whole context, this might increase job satisfaction and thus indirectly improve the working conditions of herself and others. Therefore, not only the quality but also the efficiency of the work situation improves.

Acknowledgement:

We are deeply indebted to Professor Markku Nurminen at the University of Turku and Professor Pertti Järvinen at the University of Tampere for their valuable comments on earlier versions of this manuscript. Thanks, also, to Christopher Grapes, who has helped with revision of the English language. The Swedish Cultural Foundation in Finland (Emmy and Rudolf Alfthan's Foundation) and the Foundation for Economic Education have through financial support to Inger Eriksson contributed to the accomplishment of this paper.

The design of the simulation model described in this paper is the result of cooperation. The actual division of work has been such that Inger Eriksson has done the specifications and planning, while Annika Finnäs has done the programming. Also the paper is a result of collaboration although the main responsibility is shared in the following way: Annika Finnäs has authored section 4 with the corresponding appendix while the other sections are written by Inger Eriksson. However, in the Knowledge and Work research group all work is done in a very cooperative manner. For this reason thanks are also due to the other members for their comments and suggestions.

References:

Bjerknes, G., Bratteteig, T., Kaasboll, J., Nygaard, K., Sannes, I., Sinding-Larsen, H. and Thingelstad, G. (1987). *Å implementere en idé - samarbeid og konstruksjon i Florence-prosjektet. Rapport nr. 3 fra Florence-prosjektet (juni 1986 - juli 1987).* Institutt for informatikk, Universitetet i Oslo.

Brown, G. P., Carling, R. T., Herot, C. F., Kramlich, D. A. and Souza, P. (1985). Program Visualization: Graphical Support for Software Development, in *IEEE Computer*, Vol. 18, No. 8, Aug. 1985, pp. 27-35.

Eriksson, I. and Reijonen, P. (1989). Training Computer-Supported Work by Simulation., Presented at *The IFIP WG 3.4 Working Conference, Helsinki, July 31 - August 4, 1989*, also forthcoming in *Education & Computing.*

Eriksson, I., Kalmi, R. and Nurminen, M. (1988). A Method for Supporting Users' Comprehensive Learning, in J. I. DeGross and C. H. Kriebel (eds.), *Proceedings of the Eighth International Conference on Information Systems, ICIS,* Pittsburgh, Pennsylvania, 1987, pp.195-217, also forthcoming in *Education & Computing.*

Hartson, H. R. and Hix, D. (1989). Human-Computer Interface Development: Concepts and Systems, in *acm computing survey*, Vol. 21, No. 1, March 1989, pp. 5-92.

Hellman, R. (1987). A Fictitious HIS-Reconstruct of an Information System, in P. Järvinen (ed.), *The Report of the 10th IRIS Seminar*, Part 1, Acta Universitatis Tamperensis, Ser. B, Vol. 27, Tampere. Pp. 205-219, also to appear in *Computers and Industry.*

Larson, J. A. (1986). Visual Languages for Database Users, in Shi-Kuo Chang (ed.), *Visual Languages*, Plenum Publishing, New York, pp. 127-147.

Mantey, M. M. and Teorey, T. J. (1988). Cost/Benefit Analysis for Incorporating Human Factors in the Software Lifecycle, in *Communications of the ACM*, Vol. 31, No. 4, April 1988, pp. 428-439.

Nurminen, M., Kalmi, R., Karhu, P. and Niemelä, J. (1987). Use or Development of Information Systems: Which is more Fundamental?, in P. Docherty, K. Fuchs-Kittowski, P. Kolm and L. Mathiassen. (eds.), *System Design for Human Development and Productivity: Participation and beyond*, North-Holland, Amsterdam, pp. 187-196.

Rasmussen, J. (1987), The Definition of Human Error and a Taxonomy for Technical System Design, in J. Rasmussen, K. Duncan and J. Leplat (eds.), *New Technology and Human Error*, John Wiley & Sons, Chistester, BG.

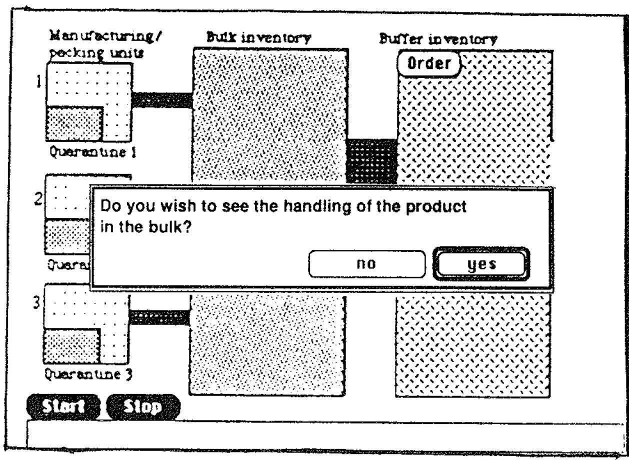
Shneiderman, B. (1988). We can design better user interfaces: A review of human-computer interaction styles, in *Ergonomics*, Vol. 31, No. 5, pp. 699-710.

Summersgill, R. and Browne, D. P. (1989). Human Factors: Its Place in Systems Development Methods, in *Proceedings Fifth International Workshop on Software Specification and Design, ACM SIGSOFT Engineering Notes*, Vol. 14, No. 3, May 1989, pp. 227-234.

Trigg, R. H. (1988). Guided Tours and Tabletops: Tools for Communicating in a Hypertext Environment, in *ACM Transactions on Office Information Systems*, Vol. 6, No. 4, Oct. 1988, pp. 398-414.

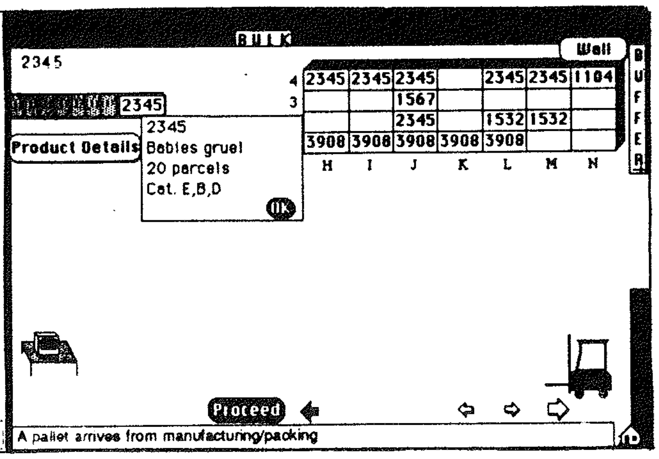
Appendix A

The best introduction to the visual simulation model is a sample session. Below a series of screens that appear during the simulation of *the delivery of a product pallet from a manufacturing/packing unit to the bulk inventory* is presented. The screen illustrations should help visualize the system while the texts supply the actions.



Do you wish to see the handling of the product in the bulk?

Screen 1



2345

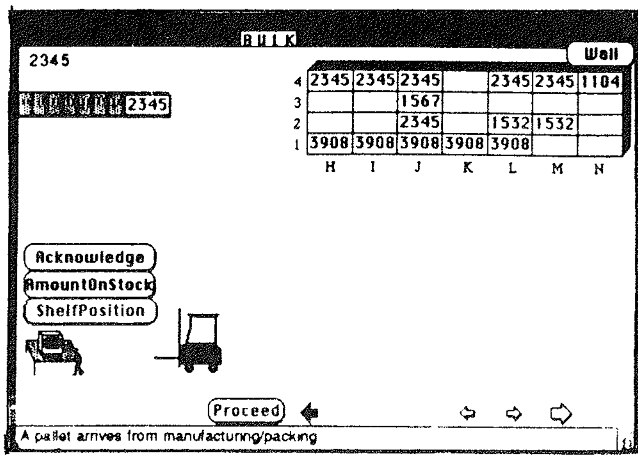
2345

2345
Babies gruel
20 parcels
Cat. E,B,D

	H	I	J	K	L	M	N
4	2345	2345	2345		2345	2345	1104
3			1567				
			2345		1532	1532	
	3908	3908	3908	3908	3908		

A pallet arrives from manufacturing/packing

Screen 2



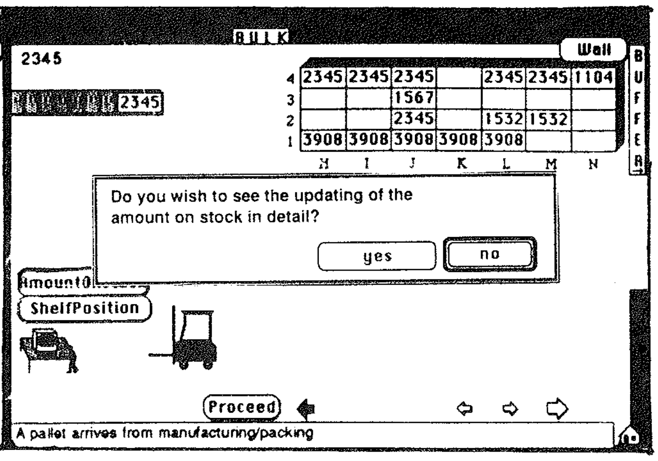
2345

2345

	H	I	J	K	L	M	N
4	2345	2345	2345		2345	2345	1104
3			1567				
2			2345		1532	1532	
1	3908	3908	3908	3908	3908		

A pallet arrives from manufacturing/packing

Screen 3



2345

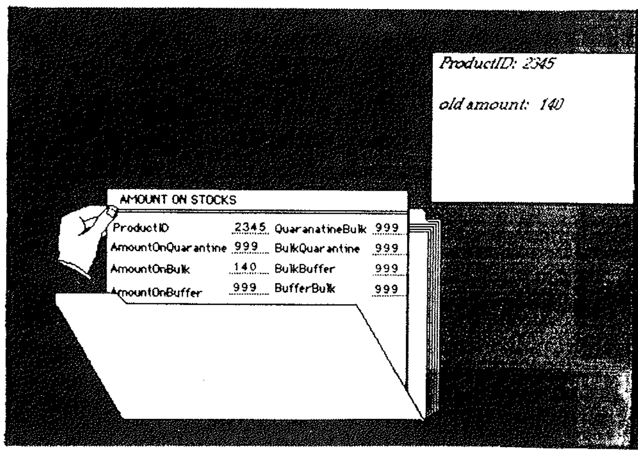
2345

	H	I	J	K	L	M	N
4	2345	2345	2345		2345	2345	1104
3			1567				
2			2345		1532	1532	
1	3908	3908	3908	3908	3908		

Do you wish to see the updating of the amount on stock in detail?

A pallet arrives from manufacturing/packing

Screen 4



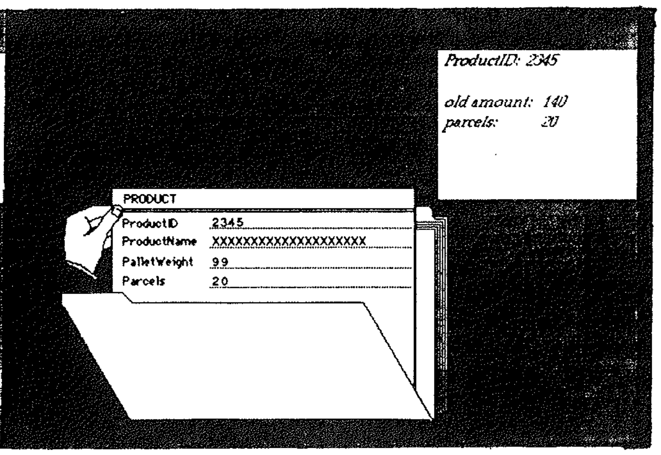
ProductID: 2345

old amount: 140

AMOUNT ON STOCKS

ProductID	2345	QuaranatineBulk	999
AmountOnQuarantine	999	BulkQuarantine	999
AmountOnBulk	140	BulkBuffer	999
AmountOnBuffer	999	BufferBulk	999

Screen 5



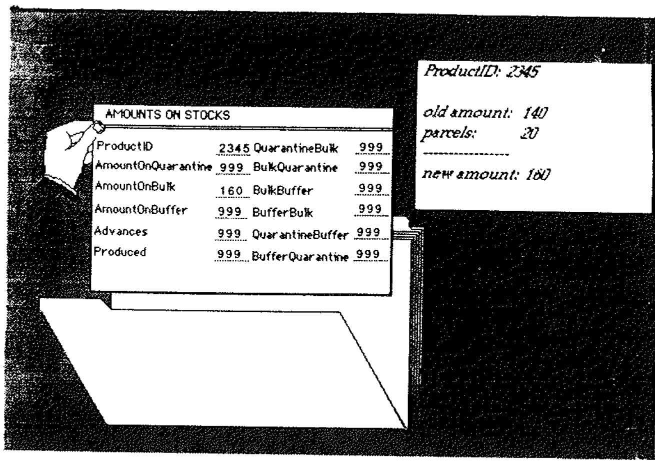
ProductID: 2345

old amount: 140
parcels: 20

PRODUCT

ProductID	2345
ProductName	XXXXXXXXXXXXXXXXXXXX
PalletWeight	99
Parcels	20

Screen 6



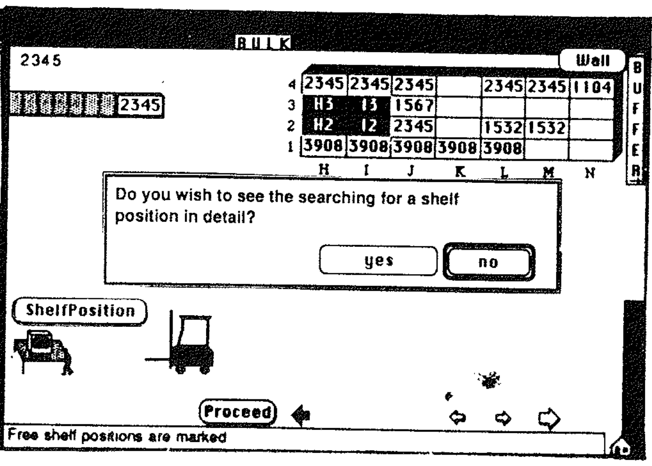
AMOUNTS ON STOCKS

ProductID	2345	QuarantineBulk	999
AmountOnQuarantine	999	BulkQuarantine	999
AmountOnBulk	160	BulkBuffer	999
AmountOnBuffer	999	BufferBulk	999
Advances	999	QuarantineBuffer	999
Produced	999	BufferQuarantine	999

ProductID: 2345

old amount: 140
parcels: 20
----
new amount: 160

Screen 7



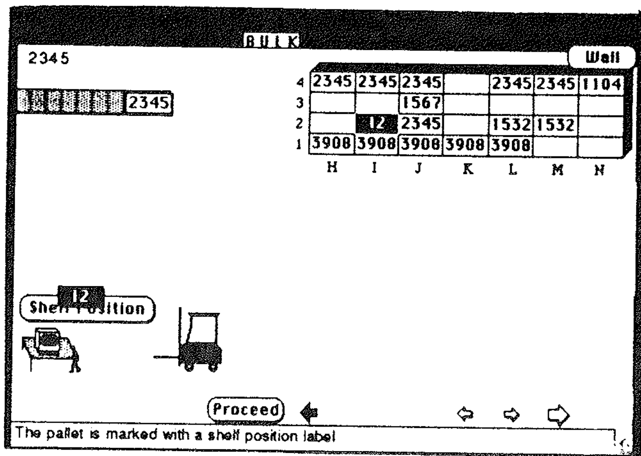
	H	I	J	K	L	M	N
4	2345	2345	2345		2345	2345	1104
3	H3	I3	1567				
2	H2	I2	2345		1532	1532	
1	3908	3908	3908	3908	3908		

Do you wish to see the searching for a shelf position in detail?

yes [✓ no]

Free shelf positions are marked

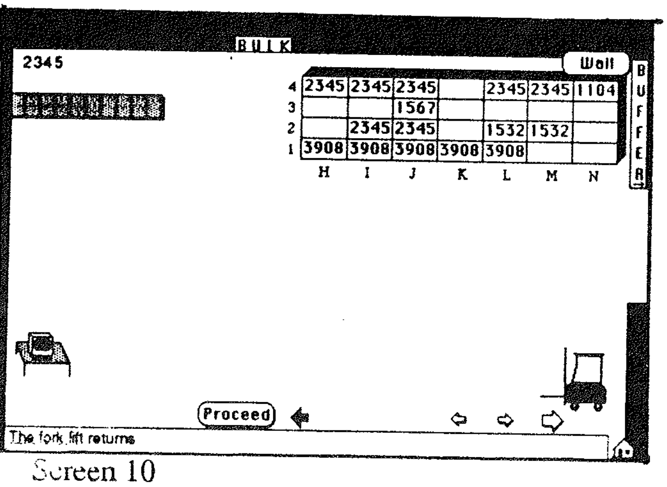
Screen 8



	H	I	J	K	L	M	N
4	2345	2345	2345		2345	2345	1104
3			1567				
2		I2	2345		1532	1532	
1	3908	3908	3908	3908	3908		

The pallet is marked with a shelf position label

Screen 9



	H	I	J	K	L	M	N
4	2345	2345	2345		2345	2345	1104
3			1567				
2		2345	2345		1532	1532	
1	3908	3908	3908	3908	3908		

The fork lift returns

Screen 10

Screen 1

The simulation is always started from the overview level by clicking the start button. An arbitrary product is delivered from either of the three M/P units and left outside the bulk inventory. At this time the object *product pallet*, that is to say, the history card for the pallet in question is created The quantity in stock is updated and the user is asked whether she wants to see the detailed manipulation of the transaction at the section level. If she chooses *yes* the scene is moved inside the bulk inventory.

Screen 2

The pallet that was left outside the bulk inventory on the overview level now lies on the conveyor belt in the upper left corner (productID 2345). The user has clicked the product details button and obtained more information about the pallet, including productID, product name, number of parcels/pallet and category information.

Screen 3

The user has clicked the proceed button and the menu (Acknowledge, AmountOnStock, ShelfPosition) is shown. The items from the menu are chosen automatically since we are working with controlled simulation.

Screen 4

The pallet has been acknowledged and the amount in stock updated. Now the user is asked whether she wants to see the updating of the stock in detail on the computer level. If she chooses *yes* she is moved to the computer level (screens 5-7), otherwise the next action is the search for a suitable shelf position (Screen 8).

Screen 5

The user is taken on a tour to search for the appropriate card in a manual file holding the amount on stock by productID in different storages. The old amount on stock is shown and this value is written down on a "piece of paper" at the right of the manual file.

Screen 6

The search continues in another manual file, the product information file, to get the amount (parcels) by which the amount in stock in the bulk is to be increased. The obtained value is written down on the "piece of paper".

Screen 7

The stock information file has become current again. The new amount in stock is calculated and the amount in stock in bulk is updated. After this the scene returns to the section level.

Screen 8

The third item, ShelfPosition, has been chosen from the menu. The search for free suitable shelf positions has taken place and the user is asked whether she wants to see the search in detail. If she chooses *yes* she is moved to the computer level and a series of pictures are shown as was the case for the updating of the amount in stock.

Screen 9

The user has responded by choosing *no* in the previous screen and the transaction continues on the section level. One of the suitable shelf positions is chosen by the system and a shelf position label is printed and stuck on the pallet on the conveyor belt.

Screen 10

The fork lift has brought the pallet to the chosen shelf position.

The scenario method
a creative research methodology?

Paper for the 12th IRIS in Skagen, 1989

Per Flensburg, ph d
University of Lund
Copenhagen Business School
Uardav. 8 G
S-223 71 Lund
Sweden

In this paper I suggest a method called *"the scenario method"* which in literary style describe what will happened if certain things are to occur. It is applied on the topic: *"Data processing during the 1990:s"*

The purpose of the paper

The paper deals with prediction of future. The purpose is at least twofold:

1. I am interested in what will happened and thus try to somehow figure it out.

2. The second purpose deals with methodology. I think the established scientific methodologies have several shortcomings in topics covered by the "Iris community". This paper is an experiment in using another methodology.

Concerning the first purpose it is quite natural that we on the doorstep to a new decade think about what might happened during the next ten years. So have I and my thoughts are presented here.

Scientific and methodological considerations

Concerning the other purpose I think it might be elaborated a little further because it might be the most important from the scientific point of view. We must discuss not only scientific methodology but also the nature of scientific knowledge.

Roughly spoken we might claim that scientific knowledge achieves its status due to use of scientific methodology. Scientific knowledge is thus knowledge achieved by using scientific methods. There are traditionally certain claims on this type of methodology:

- **Reproducibility**, which means that the investigation can be reproduced many times with the same result.
- **Objectiveness**, which means that the result of the investigation should be independent of the person who undertakes it.
- etc

Dependent on which textbook you use, there can be more and other criteria, than the above named. Also many researchers admit that it is impossible to apply all criteria to all types of investigations. The problem thus still remains: How can we distinguish scientific knowledge from other knowledge? Or do we have to distinguish?

The latter is surely a temptation and in my opinion quite possible. However, the very base for our work is the assumption that we do something that is different from other phenomena in the society. I think we must somehow make our work legitimate, we must somehow tell other people what is specific with our way of working and thinking and why our conclusions should be taken into special consideration.

A typical scientific investigation uses the following methodological steps. First we formulate a hypothesis (or a set of hypothesis) concerning the problem that are to be investigated. As a researcher I am free to choose every problem I will, as long as it is considered as a scientific problem by the other researchers in my research community. Among other things it is to be considered as an interesting problem by many researchers.

This hypothesis shall then be verified (or falsified) by certain data collection or experiments and with help of statistical methods dependencies shall it be verified or falsified. Before we do the data collection we must make a factor analysis, determining which factors that are independent and which factor that are dependent. In fact we assume a relation that can be expressed in the following form:

$$\boxed{Y=F(x1,x2,x3,x4,\ldots xn)}$$

x1,x2,x3,x4,...xn are considered as independent variables and Y as the dependent. If this are to be possible the hypothesis must be formulated in a certain way and it must be possible to identify independent variables in the research area. We must also assume that a change in one of the independent variables does not affect the other *(Ceteris Paribus).*

Many of the questions investigated by "the IRIS community" are not of this kind. In this paper I try to demonstrate an investigation I pretend to be scientific, but without use of the traditional methods. I hope it will cast some light over other possible ways of undertaken scientific investigations.

I have chosen prediction of future as topic because it is very well suited for this demonstration. And of cause, it interests me!

To predict the future - a methodological discussion

There are several way of trying to predict the future. One way is simple extrapolation, e g say that the number of transistor functions on a chip has increased with a factor 2 every year during the past 7 years (I haven't checked the truth, but it is just an example!). Extrapolation says that this will continue for the next 7 years. However, sooner or later we reach some physical limits and this is exactly the weakness of this method: If you extrapolate it too far, it is always wrong! But it might be quite useful for predicting the near future. However I'm afraid the near future in our area is about 6 months!

Another way of predicting the future is the Delphi study. In such a study several important persons, that all are very well acquainted with the area in several sessions and with feed-back between them, explore their opinion about the future. But it requires at least 20 such important persons.

The journal "Management Science Quarterly" has done two such Delphi studies during the 80:s. Probably they will make another this year or next year.

Yet another way is to send a questionnaire to a lot of people "in the field" and ask them about their opinions. If the people "in the field" are working with strategical issues, this might be a good way. You then will achieve a good impression of what people are aiming at, of their plans. However, we all know that EDP-plans are maximum 3 years, and we have a 10 year span in this investigation.

In such a long time period there will always be some entirely new things coming up. Some major breaks in the trends are likely to occur. None of these can be predicted with the methods used above, maybe except for the Delphi study.

In this paper I ought to use all the above mentioned methods. However, since I have no possibility to undertake any of them, I must do the analysis based on descriptions of such investigations undertaken of others. This will be combined with my own opinions. They are in their turn based on my experiences, what I have read, what people have told me and, of course, on my own preferences, on my own picture of the world, what it is for the moment and what I hope it to be.

I admit that their is no objective description of future (and I don't believe there are of any other phenomenon either!). My prediction will always be based on my world view, on my interpretation of what has happened and why it has happened. In arguing for the prediction of future I think the history is very important. Descriptions of past time is the base for predictions of times to come!

Basic framework and restrictions

I will restrict my discussions to systems development. This might be in contrast to the title of the paper, but since systems development is a prerequisite for the DP systems I don't think it is misleading.

We have today several trends in several areas. They have a historical background, they go in certain directions. Some might be in conflict, some might be independent and some might support each other. In this section I will identify a lot of such trends and in the first step analyze them separately. However, as the analyze proceeds it will be inevitable to look at the interconnections.

As an overall framework I will consider systems development in the middle and several factors affecting its future evaluation (fig 1). But still some problems remain. For instance: How can I be sure to cover all important factors? The simple truth is that I can't, because there isn't any agreement about what is an important factor. It depends on the basic framework, on the world view. This must be presented and known and maybe this is the most important difference between scientific knowledge and "common knowledge", **all** basic assumptions behind the science must be known. In my personal case the most important assumptions concerns my view on (wo)man.

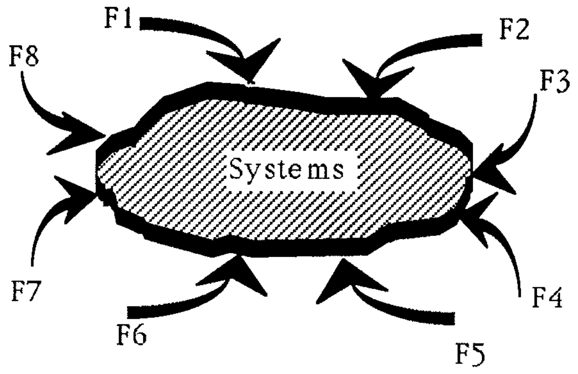


Fig 1. Model of basic framework

To summarize this very short I will say that you should respect humans, be humble, don't think you are better than anybody or shortly: Apply the Y-man view!

Now I will try to identify certain factors and continuously discuss possible relationships. First I will discuss some technical factors that might have influence.

Trends in PC:s

The overall trend is more powerful computers, a move towards work stations. We will have lots of PM, cheap and huge disk storage, powerful LAN. On almost every desktop we will during the next decade have a computer as powerful as a Vax 780.

Compatibility between computers and operating systems will increase. Concerning operative system the default standard is in the beginning will still be MS-DOS but in the late 90:s OS/2 will achieve this position. Apple will be able to run PC-programs but not the other way. Apple will continue develop it's system (it will use virtual memory e g), but maybe compatibility will be abandoned.

Unix is still a question mark in the PC-market. In the later part of the decade Unix might be one of the most common operating systems on PC:s. For work stations I think however it will be standard.

PC:s have been used for personal data processing (PDP) but in the 90:s I think they will be used for communication. E-mail will be very much used. We shall also see increased use of PC:s instead of "dumb terminals". The proposed SAA-concept from IBM points in this direction.

Work stations

They have up to now been used mainly for CAD/CAM. They will be used for DTP now. Simple and powerful graphic will attract non price sensitive users.

Minicomputers

Many minicomputer vendors have problems and show decreasing turn-around. Even DEC have trouble. One way to get around the problem might be expanding up in the main-frame area, but competing with IBM might not be the easiest thing.

IBM AS/400 seems to be market leading. It has possibilities for running Unix, but Unix is not much supported by IBM.

According to an investigation made by IDC (Computer Sweden, 1989) the market for minicomputers is decreasing and the market for LAN:s is almost exploding during the early 89-late 88. The announcement of IBM AS/400 seems to be just in time, because the S/3X customers could change to another IBM Mini instead of having to do something more radical.

Mainframes and supercomputers

I think we will have supercomputers for special purposes such as scientific number crunching, weather forecasts, advanced graphical simulation etc, but I don't believe in general mainframes There was an interesting article in Computer Sweden (May 1989) describing how the same problem (concerning air pollution) was solved in two different ways. First in a supercomputer environment and second on a PC using Lotus 1-2-3. It was the PC-solution that took the market and was considered the best.

IBM have a very strict segmentation of the market and this will probably continue (See fig 2). IBM mainframes have increased their hegemony during the latest 5 years and I see no reason why this trend should not continue. However, the market as a whole might decrease and thus will other vendors have problem. In fact we have during the last five years seen several fusions of computer manufactories and perhaps it will end with two mainframe vendors.

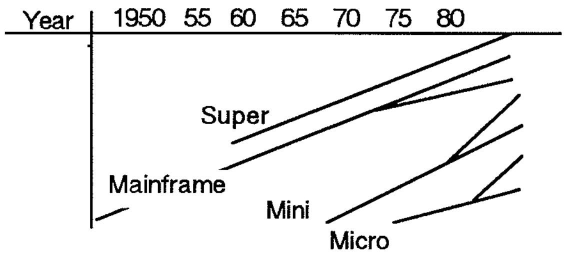


Fig 2. Different types of computers (Based on Lunell, 1985, p 58)

EDP-department

The EDP-department will probably be a company of its own and acting as a service bureau. We have seen this in many big companies and I think it will continue. The process will make the service bureau market very though and many will disappear.

This process is dangerous for the mother companies, since valuable data and knowledge about data processing will be out of control. However, at the departments local experts will arise. They will not be very educated, mainly self learned and they will have tremendous influence! The DP-department will be very anxious in employing that kind of persons. I think the traditional user-analyst dichotomy will gradually disappear, maybe not completely during the 90:s, but surely a bit of the way.

Use of data systems

I will here distinguish between three types of data systems and discuss them separately. First the most basic ones, *transaction based data systems*. I think we today in big and middle companies have computerized all transaction based data processing that are possible to computerize with positive cost/benefit. I think the development in this area will be in improving standard system. In house development will be used only for very rare and special applications.

On the other hand will the standard systems be very adoptable, yes in fact I think they will be something towards 4GL. There will be possibilities for far going customization.

Concerning the so called *MIS-systems*, which here are to be interpreted as administrative, integrated control systems based on predefined reports, I don't think so much will happened. The reports that are produced today are seldom read and I don't think any dramatic changes will occur during the next ten years.

But concerning the third type of data systems, the *support systems* (EUC & DSS), I think their use will increase. Due to powerful PC:s, easy interfaces and increasing computer literacy of the staff I think at least every department will have its own expert on using support systems (Personal Data processing, Flensburg, 1986). The anarchistic situation we have for the moment, where the companies are extremely dependent of a few key persons, will be less anarchistic during the 90:s, due to standardization of 4GL and data bases within the companies.

New types of data systems

It is always difficult to predict entirely new applications. During the 80's the spread-sheet might be considered as the great invention. In the 90's I see two candidates.

The first is due to the trend towards DTP/CAP (DeskTop Publishing, Computer Aided Publishing). The word processors will be able to do more and more CAP. This means especially use of pictures. We have today rather cheap scanners that can give acceptable pictures. In the 90's they will be better, but there will also be a need for working with the pictures. *Picture processing* (like for instance Digital Darkroom on the Mac) might be one new application during the 90:s.

Another new application might be *Hypertext*. Today almost every Macintosh have Hypercard and new applications are developed every day. Hypertext (or Hypercard) can be used for many things:

- Presentations
- Calender management
- PDP
- Prototyping
- Entertainment
- Help facilities etc

I think an environment similar to Hypercard will be the key for future PDP and also for CSCW.

System development methods

I see no need for any dramatic improvements, mainly due to the fact that there will be very little time for new development and maybe very little need too, since the PDP will take care of the most urgent needs.

Maybe we need methods for developing CSCW, but I don't think this will be actual until the later 90:s. Experience have shown a ten year delay between research and common use (c f JSP, Relational data bases).

The most dominating method will be prototyping and due to the decentralization methods for managing prototyping project will be developed. The users demand prototyping, so I think the EDP-departments must follow, whether they want it or not.

Today people talk much about CASE, but I personally think it is just another way of drawing flow charts (or DFD:s as they are called today!). They will give some technical support for systems development (e g maintaining a data dictionary) but I don't think it will be the great break through as supposed in some advertisements.

Networks

This is the key for DP in the 90:s. Almost every other application relays on a network. It is an area where the need is greater than the technical possibilities. The progress have been moderate,so far, mainly due to the lack of standards. IBM presented its token ring concept a few years ago and this became some sort of standard for mainframes and minis. However, we also need programs that can use network facilities and so far we have seen very few of them. All dependent of lack of common accepted standard.

IBM have felt the pressure and realized the importance of the market and thus presented the SAA concept. It is an integration between user interface and data communication interface. It allows you to work in the same way on every IBM computer, from PC:s up to supercomputers. From the users point of view they should not be aware of which computer they are working on.

Apple made probably a mistake when they introduced such a slow net as Appletalk. I think they will have to abandon it within a few years and introduce a much faster net. Typically it should be faster to save your files on the common fileserver than on the built in harddisc.

AI & expert systems

This might turn out to be a real flop! In the middle of the 80's there were great expectations on "the fifth generation" but these have been turned down. Many times before it has been predicted that computers should be as intelligent as the humans, but it has never been fulfilled and I don't think it will ever be.

Expert systems might be used in very limited and very specific applications and if it is supposed to replace and expert! I don't believe any human expert neither can nor will work with them. The human knowledge is far from being canned in a knowledge bank.

External data banks

I read in Computer Sweden that the people most asked for in the companies was documentalists, i e people finding information in external databases. I was a bit astonished, but it shows that information really are considered as an important resource. Obviously use of external data banks will increase during the 90:s and people who knows which information that are available and how to find it are thus very desirable.

As a consequence we might believe that the white collar worker might want to be able to do this information retrieval herself. Demand for a common interface that allows for retrieval in several data banks and maybe also demand for standardization of the retrieval procedure might grow up. Also need for education in information retrieval might occur.

Information as a strategical resource

Several studies has shown that the management realize the importance of information and information management. "The information manager" might be common during the 90:s.

I think an overall structure as shown in fig 3 will be common at least in big and middle sized companies. The common data are stored in an internal data base. From this certain standard reports are produced by the ordinary MIS-systems. It will also be possible to extract ad hoc reports with End User Computing which will also allow personal and departmental data bases. They co-operate in using PDP (Personal Data Processing, Flensburg 1986) which means support by several data systems to the individual work. I think this kind of work will increase due to decentralization.

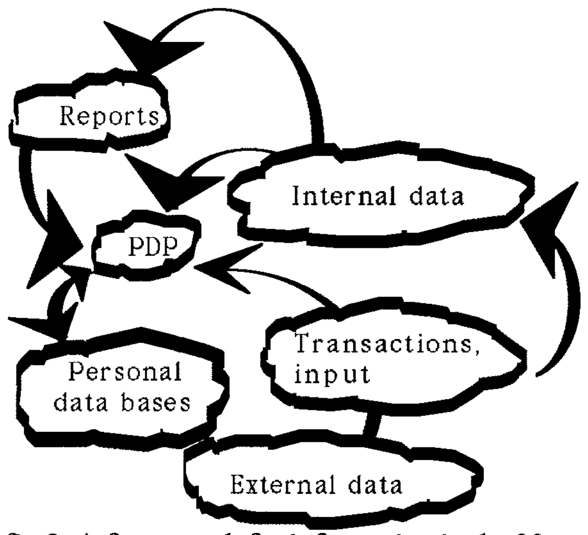


fig 3. A framework for information in the 90:s

There will also be external data banks and possibilities for integrating their information in PDP. We might also, due to increased network facilities see an enhancement of PDP to CSCW, but this will probably not occur until after 1995.

We have over the years seen a slowly development concerning updating the internal data base. The systems doing this are the so called data processing systems or transaction based systems. In the 70:s we had lots of women doing dull and tedious input work (card punching) but much of the data are now captured at the source and I think this trend will continue. A widely spread use of EAN-codes, of OCR, of EDI (Electronic Document Interchange) might automate the update process to a great extend.

The data base manager will be a key person. Maybe s(he) will become information manager instead with responsibility for the whole information management in the company.

Integration

As a consequence of the above suggested framework a total integration of the information processing in the company is needed. This means it should be possible to transfer desired data to and from any data system in the company. This is technically possible due to the network, but the data base format will probably be different. However, we have today good possibilities for moving data in certain standard format (text, SYLK, PICT etc) between different programs on the PC. We even have possibilities for moving formatted data between Macintosh and PC!

To the user it will be transparent which computer (s)he works on. The interface will be the same and the operating system will allocate tasks on different computers in order to efficiently use them all. But such operating systems does not exist today and they will probably not until maybe the late 90:s.

There will however be a great demand for this kind of integrated and standardized data processing. We see today an integration between companies working together, for instance Volvo and companies supplying parts. "Just-in-time" is a concept that will require close collaboration between data processing in different companies.

But every vendor will have a standard of his own and I think it will be very hard to achieve this total portability.

Man-machine interface

We see today a trend towards the Xerox/ Apple interface and I think it will be standard in a few years. Mouse and icons will be characteristic for the 90:s.

The desk-top metaphor might be enhanced to an office metaphor due to increased need of CSCW. We might have icons representing different rooms, different buildings etc.

Smart cards etc

Today we see increased use of cards as a way of paying, even in our every day shop around the corner. I think the magnetic strip card will survive the 90:s mainly due to the fact that we must have a lot of transactions to fill the powerful computers! There will also be no common standard for the smart card.

Some sort of intermediate conclusions

I will finish this section by summing up what I so far have discussed.

I think the great development during the 90:s will be on the PC market, due to its great volume. In the companies we have a trend towards decentralization. I think people will have it all – on their desktop! Word processing, CAP, drawing, data bases, external data banks, e-mail, conference systems - all! There will be great demand for powerful PC:s. We have today the technical capacity, but not programs that can use it. The technical evolution is far before that of the operating systems. Not until 1988 we could fully utilize the power of the IBM AT and the 80826. But only one year later the first prototypes of 80486 was presented. OS/2 was presented 1987 and very few application programs utilize its possibilities. The applications we use has a lot of facilities we never use. Think for instance of all possibilities we don't use in an ordinary word processor. Yes, the software is a real bottleneck! Maybe a wide spread use of Unix might be of some help, but it is doubtful. No dominating (read: IBM!) vendor goes really in for it!

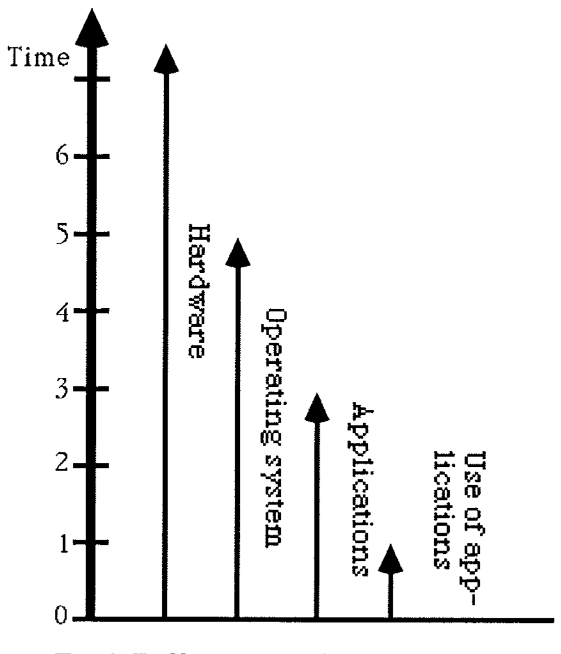


Fig 4. Differences in the evolution

Another bottleneck will be the networks. Much of the above suggested possibilities will not work unless we have an efficient network and programs that use the network possibilities! Very few do it today.

The evolution on the minis and mainframes will continue. They will more and more powerful. So will also the PC:s. And much of the processing will take place there. So strictly spoken, we might have too much computer power. However, inefficient operating systems, inefficient applications will "take care of" a part of this problem.

Creativity as a scientific method

So far I have presented an ordinary investigation. I have pointed at some important factors (variables) that might have great influence on the data processing during the 90:s. I have tried to argue about their importance and influence and have given reasons for why I think it will be so. The arguing have however, been made mainly for each factor separately. What now remains is to show that I have covered all important factors and how they interact.

Concerning the first question I can find no answer. Or rather I can never be sure to cover all important factors. I have talked to people working with forecasts (at IDC) and I have read newspapers and papers in journals. But surely some of you will find new factors I have forgotten or maybe considered having so small influence that they are negligible. However, this problem is not the great one in this section. The great problem is the interaction between the factors.

I could claim that certain factors support each other, others don't, but how can I scientifically prove it? In some cases I could use historical data (as in fig 2 and 4), but there are absolutely no guarantee for the trends to continue. Anyhow, I can do no data collection, no statistical verification and thus it is impossible to achieve scientific knowledge about the future! And still, prediction is what it all is about. Science should help us predict the result of certain actions.

It is due to this impossibility of verification I think the problem is very well suited for demonstration of what I call the scenario method.

The scenario method

I have used this method before in my thesis (Flensburg 1986), when I constructed a typical case of personal computing. It was a Weberian ideal type, and when I presented it on several companies not taking part in my investigation, I always get the same question:

- Is it sure the case is completely constructed?
- Yes, it is completely constructed!
- I could swear that it is our company, because it is exactly that way here. We have the same problems with...!

On closer examination it turned out that they recognized two or three situations among about ten. This was sufficient for a total identification with my case.
My idea is to use about the same method in this future forecast. I will describe a scenario, that I personally think is very possible (and maybe desirable, I havn't made up my mind yet!). The description should be as near real life as possible, a mock up (or prototype) would of course be the best, but lack of resources and ability!) forces me to do it in another way. I will write a short story about it.

This is where the creativity comes in. I have to make this story very near real life, as a novel. I will work exactly as an author, first gather background material (presented above) then put it all together into a nice and convincing story.

I will go into several problems. The first is if I really can write such a story. Well, I think so. The second is writing it in English. That's much worse, since I'm not very good in English (you should have noticed that by now!). But anyhow, it shold be done so here it comes.

The Day when Datator Died

a tale about a too much believing EDP-manager

Aldus Lindén, top manager of Datator Inc, opened his eyes wide. He gazed at Konrad Nelson, the economy manager, that sat opposite him. He asked with a hoarse voice:

– Is it quite true?
– Yes, quite!
– Not the slightest little chance?
– Not the slightest!
– But I can take a loan on my house!
– On 30 millions?
– Does it lacks so much?
– 15 millions in a fortnight and quite as much in the next month!
– No, that will not go. But why havn't you told anything before?
– I have! Lots of time! And you know that. But you were not interested in listening!
– Well that's right. I thought it would succeed. If we havn't got that claim for damages...
– You took a chance.
– Our last chance!

The economy manager had just presented yet an unprofitable year. The last one. Because after that it was finished, definitively finished. Paterson had said that in absolutely no uncertain terms. 1998 should be the year when Datator ceased to exist. Not even a ten year anniversary should be celebrated. It was too bloody! But how could it happened? Lindén asked Patersson:

– Have you any idea why it has happened like this?
– No, I'm no expert on the computer market. But perhaps you should talk to a such. I mean, if you intend to stay on the market.
– Stay on the market?
– Well, there will be a day after tomorrow. The fact that Datator inc ceased to exist does not mean that the whole world cease to exist!

Damned, your are quite right, thought Lindén to himself. I havn't thought about that at all. That there might be something else than Datator. Loudly he said:

– But who do you think will have a soon 60-year EDP-manager?
– Some experience can't hurt, can't it?
– But my company has gone bankrupt! Of course nobody wants to employ me after that!
– Don't say that! A bankruptcy in our days is by no means as serious as it was in older days. If you also can provide a good explanation you might earn quite a lot on it!
– Explanation?
– Well, you will of course hear from the newspapers, in the computer journals they will write about it and if you have a good and trustworthy explanation so...

Oh, Patersson you are a pearl, thought Lindén. I'm growing old and sentimental. Of course every man for himself and don't loose your head!

- Well, that sounds very good. Do you think you can keep this secret for a couple of days, maybe a whole week?
- Lindén, you need at least two weeks to find a suitable explanation! My lips are sealed!

Lindén's admiration for his economy manager increased yet more. He couldn't understand why the man had been satisfied with his comparatively modest position. He could have gone very far. But anyhow, the two weeks was to be used efficiently!

Lindén did what he always had done: He made a plan according to handbook no 3. That was his first methodology handbook, that he had implemented after only two years at Skravator inc. That was the firm that had "born" Datator inc. It happened in 1989, in the years of decentralization. At that time, God help me, everything should be formed into companies and every profit center should be a company of its own. So also the EDP-department, that became Datator inc.

Lindén had started at the methodology section at the EDP-department of Skravator in 1975 and rather fast produced that which was to be called Handbook no 3. Due to that he was appointed methodology manager after another 5 years he was head of department. After fours as that, he become top manager. Well, not bad and soon he was an ex top manager!

According to the handbook you should start by *identifying and demarcation of the function area.* Well, that should be the damned whole company! But that's no demarcation! The problem deals with the whole company! If you could demarcate the problem area, you should of course know what it was due to. And that was what he was to find out. And the method was supposed to help him. But it does not seem to do that. Well, go on in the text. After identification and demarcation you should *make a survey of the present state of the function area.* How does the routine works today and what is felt as problematic?

Lindén sighed. Damned if this worked! The method could not be applied to this application. Damned language you used! *"Function area"* what in the hell was that? And if it was the whole damned company that was wrong? *"Identify problem area"*, bah, he had never heard something so silly! Not to speak about *make a survey of the present state of the function area.* Incredible that you could according to such rubbish!

It occurred to Lindén that it was maybe *here* it went wrong! He had as an old methodologician been very anxious about following the handbook. Now he saw as a *problem owner* that it really was something else to in the middle of the damned junk than to be beside in the EDP-department. It might be because of that his idea of sticking to a strict methodology was met with so little sympathy.

Lindén remembered that some students from the university some years ago had made a survey of the company and maybe you could get some ideas from that. For sure it was Skravator that was surveyed, but Datator joined it. Some people had the opinion that such surveys was rather meaningless and that it was waste of time both to read them and being willing to help. However, Lindén thought that he many times got another perspective on things and he thought this was was not so very bad. Well, so they wrote:

"Scravifieing is a family of surface processes in order to make the surface both more beautiful and stronger. The more powerful processes increase the wear with up to 500%.

Scravifying means that you mount the workpiece in a holder that is placed in a receptacle. In this receptacle several chemical stuff is placed and through a complex geometrical movement the workpiece is treated everywhere. There are different processes dependent on both chemicals and movement patterns."

Oh yes, Lindén remembered his confusion when he first met this strange process. That a liquid rippling around in a tub moving as a drunken horse could have so great an effect! It worked, he could ensure that. Strange thing that the movement pattern could have such a great effect. Well, go on in the text:

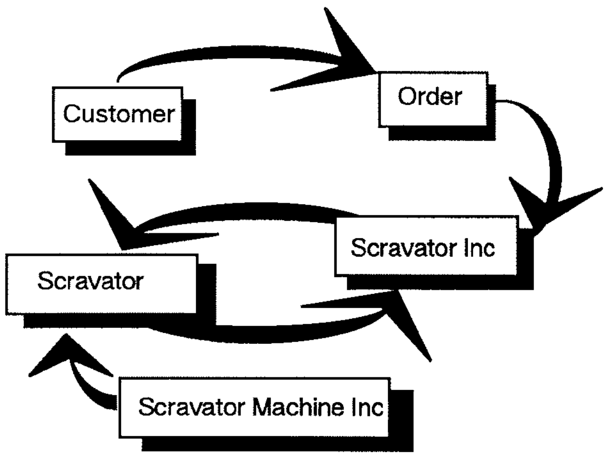
"Scravifying is very labour intense and Scravator has solved this problem in a very elegant way. They lease equipment to persons on the country, mainly farmers, that have place. These persons have scravifying as a spare time job.

Suitable orders are supplied to them. The process is mainly order controlled with few season variations"

Right! Lindén remembered his transparency. He had shown it many times. They had 5 000 people sitting in the barns scravifying. You had to keep record of both whom could do what and when they could do it. And the stuff was to be carried there and then carried back again. But the students had discovered that:

"The work organization leads to certain problems concerning coordination and control, but has that advantage that the amount of investments is low. Since the machines are considerably expensive this will despite of that give increased profitability"

Well, that machines was impudently expensive. About 100 000 SEK each. But they had a special company for the leasing of them. The students had missed that or perhaps it was not formed until later. Well, here was the section about the EDP-department:



The transparency of Lindén.

"The coordination is done mainly by computers. They had from the beginning a program developed by the EDP-department, but this has been modified later. The production manager Lars Nelson says, however, that this has not been working since they reorganized the production according to districts. Today the responsible manager for every district has to take of the necessary coordination"

Well, damned it does not worked! Suddenly 25 different districts manger should have 25 different systems without paying for them. There were only two districts still using the old system and they was to leave off in a half year. After that all connections to the old Scravator Inc was broken.

The old system was actually not *that* bad, in fact it was a good system, you could in fact do everything with it. Lindén had seen some of the home-made systems that had replaced the central system, but they were a real disappointment! Only the most simple functions were there and almost no controls. In fact, Lindén thought it was strange they could work at all. But they said so. Anyhow, Lindén thought the central system developed by the EDP-department was much better. But strictly spoken, *he* was no judge of that. That was only the involved persons. And they had lots of PC:s, especially what he at that time called "that silly mouse-toy". But they got a lot of computer power and could use very nice man-machine interfaces, that he could never provide on a main frame. And as Lindén read the other day: *A system that can not be used, is not a good system!* Well, that was as with the general methodology, it could not be applied everywhere.

Well, probably it was here it went wrong. Lindén remembered the first year they was a company of their own. At that time they implemented the organization according to districts. And of course, every district manger should have it his own way. As they said, quite right in fact: If they did not get the proper tools, how could they then be responsible for a work that demanded those tools? But Datator had no chance at all. They could not develop 25 different systems at the same time. So of course you got the reputation of being conservative and stubborn. But of course, if he had knew it was *that* simple systems they wanted, no problems. Such a system could be made in an afternoon. Well, say a week! But damned it is was not possible with an afternoon using those new 4GL:s!

It was probably a mistake to foist that standard system "Lage" on them. Well, if they had used it they way Lindén intended, it should have worked, but they did not. Obviously there was nobody knowing how it worked. And no education was wanted. It was to expensive. But obviously it was more expensive buying the system, but not using it. In fact no system should be sold without education.

Well, here was the starting point. Lindén could here himself declaring to the gathered world press:

- The basis was formed already and the creation of the company. It was done in accordance with a far going decentralization of Scravator Inc. Suddenly a development of 25 completely different systems was demanded in a very short time. Unfortunately Datator had not the sufficient resources, so there was no chance to meet the demands. They had managed to stay alive for almost 10 years, but now it was sufficient. Now they had to pay for old sins.

That was actually complaining, thought Lindén when he had tasted it for a while. The greatness of a manger is being able to anticipate such changes and manage them. The declaration above was directly disqualifying. But surely there was something in the decentralization. They havn't discovered it, Lindén had to admit that!

In fact it was a very bad decision of Scravator to get rid of its EDP-department and data processing competence. They were forced to build it up again within the company. Today Scravator manage quite well without its old EDP-department due to the fact that every single department have an expertise of its own, their own computers and connection to the common network.

The network was one of Lindén's obsessions. It was one of the latest things he had forced though as EDP-manager of Scravator. But no thanks was given. And of course, there *was* problems in the beginning. The nets was not as stable as today and above all, there was no programs that could use them. They had just began to work with mice and icons. The old Mac-interface was still going strong! At that they time they only had their standard desktops, today you have the whole company at your screen. Thank God for Hypercard, that introduced the whole way of thinking! Or maybe Bill Atkinson instead?

Lindén was nostalgic. It was not until 1987 they bought the first Mac at the computer department. It was a new secretary that absolutely refused to write on their PC:s, despite of the fact that they had Windows and the whole thing. And after a while the girl was complaining about not having her machine for herself. The whole damned section wanted to work with it! So began the problems with making apple and bananas speaking the same language without changing their shells. But they had succeeded! 1990 was all Mac:s and PC:s on the same net and if you used Word, Wordperfect, Excel or Pagemaker the whole machine environment was transparent. All was stored on the 2 Mb fileserver.

It was however a little bit slow at the beginning. But after installing the new Ethernet, based on optical fibres, it began to speed up. The latest was that they had started to use the net for intern television. Yes, in fact they had a video camera attached to the computer and used it as a scanner. At a scanning frequency of 20 pictures/sec they obtained animated pictures. The new 75 MHz computers managed that quite well. The pictures was then send via the net. Well, that was the e-mail of today!

Lindén remembered installing MemoCom, a combined e-mail and conference system. But they called it "Group Decision Support System" and has absolutely not intended the resulting huge propagation. But in fact, it was quite ideal for mutual contact to all 5 000 scaravtors!

But that does not helped Datator. You get paid for the installation, but that was all. And no other company was interested. Besides it was a child's play to do it in the new Ethernet IV.

Datator had been one step after the whole time. You executed the task you were given by Scravator, but was never given the chance of coming with initiatives of your own. It was due to their information manager, a systems scientist and a master of business. extremely smart and efficient. And pretty too! She took the breath from you, that Pernille!

It was due to her Scravator managed its complex information processing. Lindén had just been her tool. He had learned quite a lot and if he now was appointed Information Manager he could surely do a good job. But what's the good of that now?

Darn, that was the way of doing it! The winding up was already planned, but due to fast development, both of the technique but also of Scravator, it came maybe too early. They intended to wind up themselves, but had had now time for planning it. The network and the whole staking of making different computers speak to each other shows the immeasurable foresight of Datator and especially by its manager! Hmf...

But was Datator not able to get other customers and that way continue to exist? That was a good question and Lindén could not answer it at once. Then the telephone rang:

- Hallo, Aldus, it's Pernille!
- Oh, hallo! What gives me the honor?
- Well, it is actually quite sensitive and I don't want you to tell anybody, but I have an offer from another company and I will probably abandon Scravator.
- Oh, how sad! It's due to you the company works!
- Oh, you are flattering me! You don't mean that!
- To tell the truth, I was just thinking about it, and I really think that without you company had simply not managed its extremely complex information processing.
- You are kind, Aldus, but now I will tell you another secret. Do you know why the information processing works?
- No, tell me!
- It is because the people can decide themselves!
- The people decide themselves? But you have to supervise them! They can make wrong decisions!
- Uhu, do you think you can make better?

Lindén was silent. It is said that it is impossible learning old dogs to sit, but at this very moment Lindén went through a mental transformation. He suddenly realized all that had been wrong in his work so far. He had believed that he was superior to other, that he knew how things was to be carried out. He had been stubborn, self assured, high-handed and pompous! He had...

- Hallo, are you still there?
- Uh, sure. You have a damned good point there! As old as I am I havn't thought that way at all!
- You havn't?
- No, and I have to pay for that now!
- Why?
- Well, if you can keep for yourself I will tell you a secret.
- My lips are sealed with seven entrances!
- In fourteen days Datator goes bankrupt!

There was a long silence. At last Lindén had to shout:

- Hallo, are you still there?
- Yes, I was quite struck dumb and amazed! What do you intend to do after that?
- Oh, surely something will turn up!
- But you have not anything in mind?
- No, and God knows if there ever will be!
- Well, you might be information manager at Scravator Inc. It was actually because of that I called!

Lindén almost cried. It was too much! First this tremendous insight that he for almost 40 years had had basically wrong and now this chance to correct it. As in a dream he heard far away:

- Hallo? Hallo? You are still there Aldus?
- Yes, yes I was quite overwhelmed!
- Yes, I think you are almost crying!
- I am!! I am!! I can't stand it!! I thought life had come to an end!!

Uncontrollably sobbing from happiness Lindén put the receiver down. A reborn 60-year old man!

What to do next?

Suppose we have a good, vivid and literature oriented scenario description. What shall we do with it? And why? I think these are questions that are to be answered at a general level.

If we read a good novel, we might get a very good impression of what is happening and why it is happening. The description will also contain tacit knowledge to a certain extend. Thereby the description will be much richer than the traditional scientific analysis such as bar graphs, correlation schemes etc. It is also more personal and it might be more convincing.

The latter points at another important aspect concerning scientific knowledge on the whole, that is the reasons for doing it. I think it must be based on a personal ethical conviction, a personal standpoint that this is something that will be good for mankind. It must be based on an explicit world view and that's the great difference between scientific knowledge and "ordinary knowledge". The scientist must be very explicit about why (s)he does what (s)he does.

In this case I have written something like a "Science fiction short story". It describes a scenario I personally think will be favorable and possible to realize. I tried to identify some of the forces that encourage this development and also some that suppresses it. If my story is good and if it will be read be a number of influential people, it might be some sort of self fulfilling prophecy, and if that will be the case I'm very much satisfied!

References

Computer Sweden: A lot of numbers from 1988 and 1989, where different people have belief's concerning the future. Especially several investigations from IDC.

Flensburg P: Personlig databehandling - introduktion, konsekvenser, möjligheter, Ph D thesis, Studentlitteratur, 1986

Lunell H: Datalogi - en inledande översikt, 2 uppl Studentlitteratur, 1985

"IT´S LIKE WALKING IN SYRUP" - a participative change process

© Tracey Golrang and Ann Hägerfors
Applied Psychology, Lund University
Information and Computer Science, Lund university

Abstract: In this paper we present an explanatory model which shows how individual and groups perceptions of organisational changes play a predominant role in the resulting consequences of these changes at the organisational, group and individual levels. The model thus consists of three different dimensions, the individual, the group and the organisational. In each dimension exists the variables formal structure, information technology, tasks and actors. The key to success when change is needed is that it should be based on the active participation of people from all three dimensions who can participate on equal terms. We discuss the consequences of suggested and/or implemented changes which have been initiated either by management or by individuals and groups in the short, moderate and long run. Results from an empirical study in a large Swedish company have been used as one of the basis for the model.

1. INTRODUCTION

Our research has been conducted within the research-project "Forms for Responsibility and Participation in Continuous Systems Development" (Nissen, Sandström & Ekvall, 1988), a 3-year project financed by MDA[1]. The project is a cooperation between researchers from Information and Computer Science and Applied Psychology.

In the model presented in this paper we discuss the consequences of suggested and or implemented changes which have been initiated either by management or by individuals and groups in the short, moderate and long run. With the help of our model we show that change initiated in only one of the dimensions is met with resistance in the other dimensions and initiates compensatory or retaliatory changes.

[1] MDA stands for Människor, Datorer och Arbetsliv (Humans, Computers and Working Life). It is a multidisciplinary research programme jointly funded by the National Board of Technological Development and the Work Environment Fund.

Our model is partly based on the approaches to organisational change made by Leavitt (1965) and Kotter (1978). The model is mainly the result of our efforts to structuralise and explain our findings from an empirical study we made in a large Swedish industrial company. Our model illustrates the consequences of change within the studied organisation by combining parts from both approaches with the interaction between the individuals, the groups and the organisation. We will in the near future carry out further research in other organisations to test if this model is a general model.

It is our ambition that the model should be used as a framework to better plan the consequences of change considering the different variables/element in all three dimensions. The model does not induce any limitation in use of specific methods, any methods of measurement can be used. Our concentration on the variables formal structure and information technology in this paper is due to the study we made and the major changes made in just this company, not due to any specific limitations in the model.

2. THEORETICAL FOUNDATIONS

Leavitt (Leavitt 1965) views industrial organisations as being complex systems in which at least four variables are especially predominant; structure, technology, task and actors. Structure refers to the roles and relations which the organisation is built around; systems of communication, systems of authority, and systems of work flow. The tools, machines and techniques which help the organisation to produce it´s products are seen as the organisations technology. With task Leavitt refers to the production of goods and services, including the large number of different but operationally meaningful subtasks that may exist in complex organisations. Actors refers chiefly to people, and also to specific human knowledge and experiences which is necessary for running the organisation. Leavitt demonstrates that these four variables are highly interdependent. This means, according to Leavitt that a change initiated in one variable usually results in compensatory or retaliatory changes in the other three variables.

Kotter (Kotter 1978) presents an integrative model of organisational dynamics. In the model Kotter divides the elements in the organisation into key organisational processes and six major conceptual or structural elements. The major elements are the external environment, employees and other tangible assets, formal organisational arrangements, the social system, technology and the dominant coalition. Time is an important aspect in the model. Kotter separates between the short run, the moderate run and the long run. In the short run the focus is on cause-effect relationships between each structural element and the key organisational processes. The relationships among the structural elements and alignment among these are the focus in the moderate run. Adaptiveness is the key word when discussing the organisations future in the long run. In order for an organisation to function effectively Kotter states that an understanding of the organisations dynamics in the short, the moderate and the long run is equally necessary.

The organisation´s changing needs are usually interpreted by management. This is because only they have access to the required general information of the whole organisation. If these needs can not be satisfied within the existing organisation, management initiates changes which result in an organisation which is more compatible with the new needs. Both Leavitt and Kotter provide management with tools to predict the consequences of changes, and to help them plan and manage change. What neither Leavitt nor Kotter discuss is that it is the individuals' perceptions of a situation which governs the specific consequences of change. They fail to emphasise the individual and group perspectives. How change is experienced might not be important to management in the short run and sometimes even not in the moderate run but is certainly crucial in the long run.

The importance of focusing on individual and group perceptions of change is discussed by Maier&Verser, who state that the negative results of resistance often become apparent only after the changes have been made, they are discovered too late for correction. Thus, high-quality decisions fail because people are hostile or apathetic (Maier/Verser 1982). It is peoples' reactions to change which will determine whether a change will meet resistance; cause compensatory or retaliatory changes or be "successful" in the short, moderate and long run.

3. THE MODEL

One main assumption we make is that it is individuals' and groups perceptions of change that determine how they will act and react and that this in turn determines the consequences of change.

We analyse the consequences of changes from three different perspectives, which can also be seen as different levels of analysis or different parts constituting the organisation. These are the individual, the group that the individual belongs to in his or her work, and the organisation that the individuals and groups belong to. Each of these perspectives are divided into four parts, information technology, formal structure, tasks and actors.

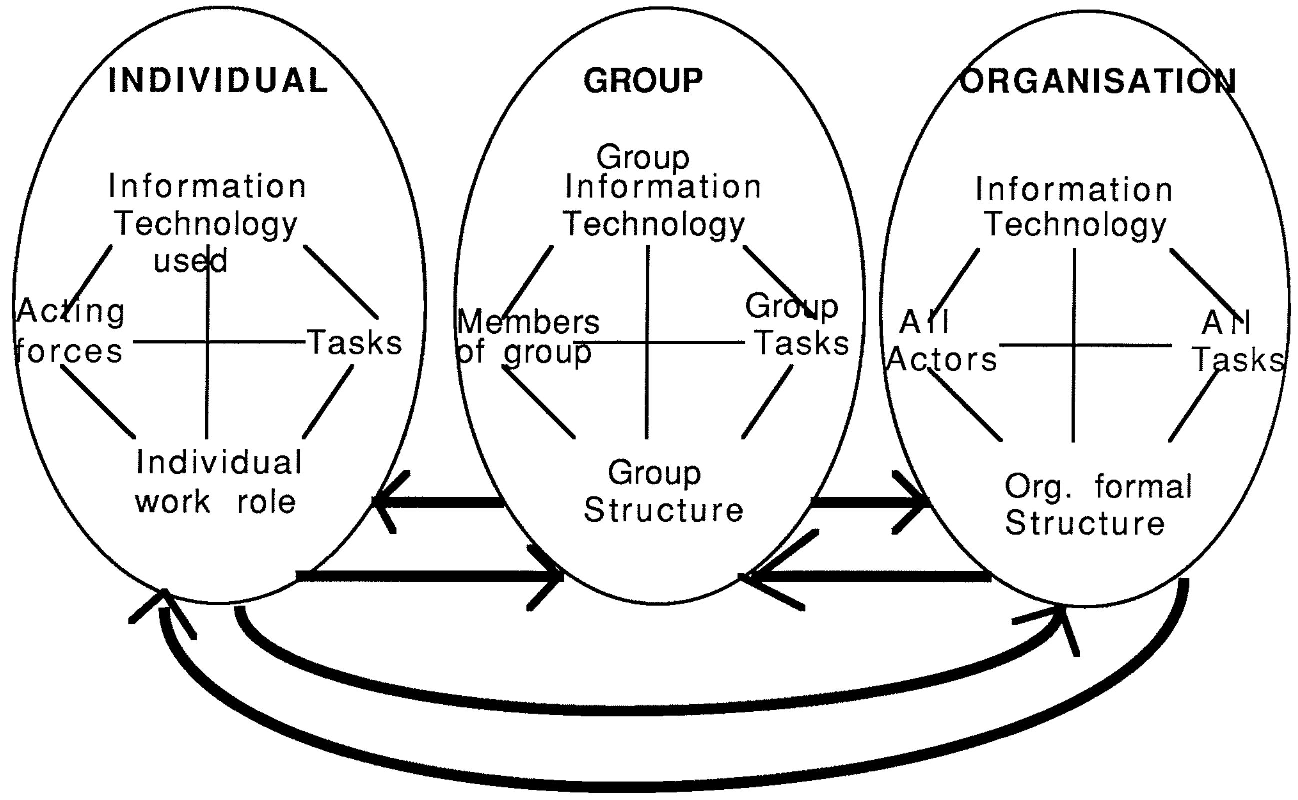


FIGURE 1. The three levels of analysis. The arrows show how a change is initiated in one level and how people´s perceptions of that change causes consequences in every variable in each level.

In our model the organisation is constituted by everything that comprise the company, in short what the organisation does and why it exists, it´s raison d´etre. We consider formal structure, information technology, tasks and actors in the organisational dimension, as well as in the group and individual dimensions.

The concept formal structure we define in Kotter's terminology as being composed of "all formal systems that have been explicitly designed to regulate the actions of an organisation´s employees (and machines)." We also include leadership styles and organisational policies.

Information technology is limited to only information and computer technology and in particular the development and use of information- and data systems. We use *information system* as a concept meaning a system that includes a computer system, a data system and people. By *data system* we mean an application, in this case a material planning programme, and a computer system (Checkland 1984, Eriksson 1986, Flensburg 1986, Jonson 1987, Wood-Harper 1985). We see systems development as a continuous process. Therefore we do not separate between systems development and systems maintenance. We agree that "The completed information system must not be looked upon as an invariable product." (Sandström, 1985). In our empirical study we have had a system in use as our object and thus as far as information technology changes are discussed we will in this paper concentrate on what is usually called systems maintenance.

We use the same definition of tasks as Leavitt does. Informal social structures, peoples knowledge, experiences, value systems and perceptions we refer to actors.

In our model the four concepts do not have the same practical content in the different dimensions. The contents are tied to the dimension, that is to say that when we speak of e g group information technology we mean the information technology that is used by the group and not all the information technology that is used by the organisation. Neither do we speak about the information technology used by a single individual.

A change made in an organisation affects directly or indirectly all three dimensions and each variable within each dimension. The consequences of the original or subsequent changes however are not the same in each dimension. Because of this the change can create a misalience between the three dimensions and the variables within them. This misalience may not become obvious immediately, but can be devastating in the moderate or, above all, in the long run.

Our model will be enlightened with results from the empirical study. Both management initiated and group initiated change took place. We describe and analyse, with the help of our model, the consequences of structural and technological changes, which had been initiated by managers or by the group, on the three dimensions: organisation, group and individual and the four parts within each dimension.

4. THE EMPIRICAL STUDY

Our study was conducted in a material planning department of a large industrial company. During the investigation we studied both the company as a whole and the department. The study within the department was dispositioned in cooperation with the material planning department as follows:

1. introductory meeting
2. questionnaire
3. meeting to discuss the analysis of the questionnaire
4. interviews with each of the employees
5. presentation of our results of the study
6. if change process needed, start of that here

Our disposition was revised after the meeting where we discussed our analysis of the answers to the questionnaire. The employees found that after seeing their situation "blown up on the walls" (since we presented our analysis on large papers clipped on the walls of the conference room) they felt a need to try to start a change process at once. What they did was to form a biweekly meeting in which they discussed which things they should give priority to pursue, decided on strategies on how to pursue them and which people to invite to explain things they felt a need to know about (for a detailed description of points 1-3 in the investigation see Golrang and Hägerfors 1989).

The empirical information used in this paper has been taken from the questionnaire, the interviews, the departmental change meetings which we attended, meetings with different managers and computer personnel and from certain company documents and an observational study (Brattgård and Ingman, 1989).

Company

The company in which we conducted our study is part of a multinational organisation. Our data mainly refers to the Swedish company. The Swedish company structure was built up around the traditional hierarchical pyramid, where decisions are made by management in the higher levels of the company and are carried out by a specialized labour force.

Department

The material planning department handled inventory control. When this investigation was carried out there were nine people employed in the department. Seven of these employees were material planners and the other two were delivery control clerks. One of the material planners acted as the group supervisor. The supervisor had held his position for a year. He had not previously worked with material planning within this company. He was not only manager for the material planning department but for a number of other departments as well. The data system they used was bought from another company. It was adapted to the needs of the studied company. The system included routines for material planning, purchasing and handling of orders. The material planning department used it to do purchase suggestions, make queries, planning and part of delivery control.

The most important aims in introducing the system were to improve service to customers, have less capital tied up in stock and to make the work easier for the material planners. The system was not to do their jobs for them and they were not to work with the system all day. It was to give the material planners decision support. It was to provide an aid for control and simulation. The system was taken into use in the Swedish company in 1986.

5. MANAGEMENT INITIATED CHANGE

The Swedish company had undergone major management initiated changes in recent years. These changes had taken place predominantly in the variables structure and information technology. Recent changes made within the company had not only been directed towards continuing a traditional structure, but if anything had placed even more power in the higher levels of the company. This however stood in conflict to the expressed company policy which stated that no one should be forced to accept and use the new machines, routines etc.

The company had been structurally reorganised several times since 1986. Divisions were fusioned, departments were transferred from one division to another, new departments were established and transfer and appointment of managers was frequent. Another change was that the computer department recently was separated from the company and turned into a subsidiary. This entailed that the computer company was to serve the entire international organisation instead of giving computer support only to the Swedish company.

An information technology change in the form of an expansion of the use of information technology had taken place during the last few years and the company was now considering a major change in the information technology and changing to a single computer supplier. Until that time they had three different suppliers of hardware. It was an expressed policy within the company to develop flexible data systems, but in reality it was usual that no further development took place after implementation of a system. We consider this to be a maladaptive behavior. The adaption of a system to a continuously changing environment and conditions determines how effective the system and use of it will be.

Most traditional systems development methods have the view that systems development is to be management initiated (Bansler 1987). Systems development in the company was carried out in accordance to this view. The computer department received directives from management on the projects they wanted carried out. A project-group was formed which worked within the frame. Usually the project-group was composed of people from the computer department, AD people and one user representative from affected departments.

A data system is just one part of an organisation and has to fit into the organisation. We believe that the systems development must not be reduced to the construction of a data system. This was not considered in the company. Systems development is however still strangely enough, like in this company, carried out as if the technology was the determinant factor (Hedberg et al 1987). But the fact remains that people are not likely to support changes when they are not allowed to participate in decisions concerning what changes are to be realised (Leavitt 1965).

When traditional systems development methods are used it is likely to find a lack of real involvement on the part of the future users (Hägerfors, 1985) and this was the case in the system we investigated. Many traditional methods mention the importance of some kind of user participation in the development process. This participation is however very often not sufficient and was not sufficient in this company.

Newer approaches to systems development discuss real user participation and the requirements that have to be fulfilled in a much more extensive way. Users, who are experts in the application field and expert in their jobs, must participate through the entire development process, all categories of users have to be at least represented, participants in systems development projects have to be relieved of their ordinary work etc (Bansler 1987, Checkland 1984, Eriksson 1986, Flensburg 1987, Hugoson et al 1983, Mumford 1983, Wood-Harper 1985).

5.1 CONSEQUENCES FOR THE ORGANISATION

The changes initiated by management were implemented and caused consequences, i e disturbance and resistance, high turn over, dissatisfaction, non-following of rules etc. Repeated reorganisations became necessary due to management not planning and foreseeing the secondary consequences of change in the three dimensions. The restructuring of the company both caused a misalignment between the tasks carried out in different departments and affected negatively the actors in the company. Numerous such undesirable consequences still existed in the organisation.

Departments which work together were placed in different divisions where the division managers did not focus on the same economic variables. This management initiated structural change thus aggravated departmental cooperation. Tasks from certain departments were transferred to new departments thus adding even more levels and structures to the already complex company structure and resulting in an even more specialized labour force.

The restructuring within the company caused a turbulent atmosphere. The employees did not know what would happen in the future. They felt they did not even know if the international organisation´s directors would close down the Swedish company and move it abroad. According to one employee many employees had already left the company or transferred to new departments within the company and that the company had increasing difficulties to recruit personnel.

Implementation of the material planning system was a technological change by which management tried to obtain a higher degree of efficiency, e g a change in the organisational structure. The goal was not fully achieved. The company attained a higher degree of service to customers by giving better decision support. The time spent on concluding a purchase was however not reduced. The goals of tying up less capital in inventory and raising inventory turn over was not fulfilled, but according to the new policy of stressing service to customers that goal was no longer important.

5.2 CONSEQUENCES FOR GROUP

These management initiated structural changes in the company induced consequences in departemental structure, information technology, tasks and actors. Within the department the structural changes influenced the tasks of the group, it lead to diminished work content. Several tasks were moved to other departments. Division of work tasks between the employees in the department increased.

The administrative development department that was involved in the material planning systems development was later placed in another division than the material planning department. A new AD department within the division was established and the material planning department became connected to it. The personnel in the new AD department did not know about material planning or the data system. This caused problems when the material planners needed changes on or had queries about the system. The largest problem however was the difficulty to obtain changes. The employees in the department had a lot of improvement suggestions. They felt that these were largely ignored by the new AD department and by the computer company.

This structural change caused the material planners to use the system in a different way than planned. They were not aware of and had nobody to tell them about all the possibilities of the system, they tried to use the system as little as possible, they found ways around the routines of the system etc. A representative of the "old" AD department said "In practice the system has induced a lot of routine work and the users have had no time to use the more sofisticated parts of the system. The users do not see the computer as an aid, a tool but as a slave driver."

The new manager within the department was neither knowledgeable about the material planning system nor about the work carried out in the department. They therefore received no support from him. This meant that the group only occasionally discussed their work with him. He was forced to promote one of the material planners to supervisor; the supervisor knew more about both system and work than the manager.

The group stated that on numerous occasions the manager asked their opinion about a specific question. This they believed was pointless as management already had made a decision. Their opinions were never listened to. This meant that the decision making routines in the company did not allow the labour force to influence the changes that they were expected to carry out. If anything, asking employees their opinions without paying attention to them when the decision was made added to the turbulent atmosphere and dissatisfaction with company and management.

Many of the factors mentioned above plus the turbulent atmosphere within the company caused the department structure and routines to be perceived by the group as being unstable and chaotic. They stressed the need of having a stable department and that "To remove the bad atmosphere requires a manager who is knowledgeable in the routines in the department and is sure of himself." The turbulent atmosphere within the department led to that not all new employees in the department received satisfactory introduction training. People did not have the peace and quiet necessary to learn.

The group believed that delivery control should belong to purchasing rather than material planning. "Then they would see how many of their fine suppliers did not supply in time." They also stated that it was difficult for material planning and purchasing to cooperate in that purchasing is steered by economy whereby material planning is controlled by level of service. The separation of these two departments to different divisions was a catastrophe for the cooperation between them. This structural construction had strong negative consequences for the material planning department in that they were used as scape goats whenever deliveries failed, even though they had no means to influence for example choice of supplier.

The new information technology in the department, the data system, was not supposed to rule the work of the employees and it was not the intention that they should be tied to the terminals in their work. The system was to be an aid to them in their work. It was used, however, as an essential element in nearly all tasks within the material planning department. The department would no longer be able to function without the aid of the data system. Most of the employees felt that they spent seven eights of their work day in front of the terminal. The implementation of information technology in the department thus affected both tasks and actors within the group.

One of the persons who worked in the department before the system was implemented expressed a strong wish not to be so bound by the system. Several of the more recently employed also felt that "... the system rules the work, we cannot do any work at all if the system is down.". All of the employees felt that since they were so dependent of the system the very long response time was very irritating. Other problematic things were the dependency of the terminal and computer security measures which stopped them from getting all of the information they needed to do their jobs properly.

Before the system was installed material planning was done manually. All the material planners knew and understood the model used for planning. After three years of using the system, only the "veterans" in the department had an idea of the model used in the system and even these persons have forgotten much of the model. "It is a shame that I have forgotten all the formulas and how to work manually. You do not need to know this anymore as the computer does it for you. The new employees have no idea about which formulas the computer uses." The technology change diminished the professional skills in the department. This might cause that people applying for a job in the department have less education and that knowledge which is valuable to the company disappears. The employees will become experts in using the system, not experts in material planning. This will certainly have subsequent consequences for future systems development and the quality of future information technology changes.

5.3 CONSEQUENCES FOR INDIVIDUALS

All of the persons in the department had strong views on the changes initiated by management. We enlighten these views with some quotations from the study. The quotations are self explanatory. Each individual stressed different aspects of the restructuring and new data system and how they have influenced just *his or her* work and work environment. We will comment on the general consequences of these remarks.

"Everything now revolves around the supervisor. When (earlier) a problem arose at work it was something good because it was a change and fun to try and solve. Now such things are transferred to the supervisor as soon as they arise so that he can keep an eye on them. When this happens it feels as if people do not think you are capable to deal with it."

"In the beginning the work involved a lot of variety but it has narrowed a lot over recent years and become monotonous. You need variation in your work"

"You can be proud of e g the achievements of your children. It is however to big a word to use in connection with my work. I will not use the word 'proud' for my work, but neither am I ashamed of it."

"The first two years I thought the job was incredibly interesting. It was an honest company to work in and less hostilities in the company. But then a lot of disturbance and insecurity has been brought in through all the reorganisations. Now I do not think the work is so interesting any more. Everybody has stagnated in their way and have become permanent fixtures. You try to resist but you cannot go against the wind all the time. You have to adapt."

"At times I do not want to go to work. The atmosphere is not secure and the wage is low. They move people around here and there without actually asking them what they think."

"The division of work between material planners was introduced only as it made work easier for the purchasers. It would be of more value for us to have a different division of tasks."

"I feel that my job is independent but that is becoming more controlled. I feel that the increasing amount of statistic is one of the factors which is controlling my work more and more."

"My enthusiasm for the work is not so great today as it was when I applied for the job. I am dependent on the computer to do my work, the work has reduced in status, one is only a key board pusher now.

"If the amount of terminal work continue at the same rate it is not possible to do anything about the lack of variation in my work." "I am now dependent on the computer and is unable to work when the computer is down. I think that the system is very slow."

"My job is monotonous and I get tired of pushing the key board all day."

The changes in company structure and technology reached as far down in the company hierarchy to an individual´s own work tasks and attitudes. Both the structural changes and the data system resulted in that an individual felt that work tasks became monotonous, narrow and meaningless. They were not satisfied with the department structure, the company structure or the technology in that all controlled their work and work environment to a high degree. They felt that the data system was clumsy and slow. This added to their general irritation and dissatisfaction with their work. Generally, the individuals job satisfaction and motivation have gone down after the structural and technological changes. The consequences of this might be that people start looking for other jobs.

6. GROUP INITIATED CHANGE

Changes are important to people and groups of people that are expected to change their work and work-habits. Individuals and groups should be given the chance of not only carrying out the previously decided changes but should, both for individual and organisational well-being, be able to influence the planning and decisions involved in the change process (Ekvall, 1988).

It has been shown in numerous empirical investigations that organisation, groups and individuals benefit from a participative system. Organisations are stated as benefiting in that employee turn over is low, absenteeism low, high quality of work (Goodman et al 1982) and that change is easier to carry out in participative systems in that,"There is a general agreement,too, that loyalty to purpose can be achieved through participation; that people will support what they help to create'." (Leavitt 1965). In practice the decision of whether employees work will be changed by the implementation and use of a data system must also be made by themselves (Nissen 1976).

Individual job satisfaction has been shown to increase with the degree of trust that subordinates feel their manager have in them, and the degree to which there is participation in management decisions (Ritchie and Miles 1970). Other investigations have shown that if an individual is able to control and influence his own environment stress can be modulated (Johansson 1981, Lundberg and Frankenhaeuser 1978, Karasek 1981). The importance for an individual to have a moderately varied inflow of stimulation has also been found to be a modifier of stress (Frankenhaeuser and Johansson 1981, Zuckerman 1969). It is our belief that both of these psychological elements could be satisfied within an active participation in work and system development.

As a result of the research carried out in the department, the employees in the material planning department decided to try to change their work environment, their data system and their relations to other departments which they needed to cooperate with. Management initiated changes had caused a lot of misalliances in the three dimensions. These misalliances were not seen by the individuals working in the company. The consequences of the misalience were however felt by everybody and were expressed by individuals as reduced job motivation and satisfaction and the company was perceived as being chaotic, unstable and turbulent. The group we investigated tried to reduce their negative feelings by initiating a change themselves. Their initiative to start a change process can be seen as perhaps the most major consequence of the management initiated changes in the company. They formed what they termed a "change meeting" which took place one afternoon every other week (see section 4. The Empirical Study).

The group decided to begin the change process with trying to improve their position in the company structure and their technology. They were going to try to obtain a better cooperation with the purchasing department in order to be able to do their job in a better way. The design of a model for visits to deliverers which was to be the common model for various groups was going to change the company´s structure in a minor but important way. The technology change was that they would try to get a better key board and desired changes in the data system.

All the changes the group at that point decided to give priority to were changes that required the cooperation of company authorities outside the department. They were at that time encouraged by the statement from the computer company that any reasonable changes would be realised.

All in all, the material planning department had seven change-meetings. They failed to achieve any of the changes they tried to realise. They decided it was impossible to reach any changes this way. Instead they began to have a shorter meeting every monday morning. During this meeting they were going to discuss events of the coming week, problems in work routines etc, e g they changed their focus to the internal group and individuals; things they *were* responsible for and *could* change. We, the researchers, were not asked to participate in those meetings.

6.1 CONSEQUENCES FOR ORGANISATION

Before the investigation started, management stated that the company was as one big team working in a family atmosphere and stood for competence development for the employees.

They thought the ideas that the research group represented fitted well in with the company policies. However, in reality when management began to observe the results of the group initiated change process we found that they resisted such change. They felt that they had various valid reasons why the suggestions could not be realised. Comments such as "change takes time", "we have specialized departments who are responsible for suggesting changes", "it is a good thing that the system rules their work" were frequently spoken by management towards the end of our investigation.

The result of the suggestions to change made by the group were that the management felt that the group, with the aid of the researchers, had become a nuisance. The material planners did not accept things as they were any more. The management within the company had already began to discuss a new major change in information technology. The decision to introduce this new technology had already been made, even if the material planning department had not yet been informed about this decision. The changes suggested by the group were not in alignment with this decision. The relation between the group and management thus was made worse. One other result of management resistance to the group suggested changes were that the research-group was asked to leave the company.

In the short and moderate run the group initiated change suggestions, in that they were stopped, had no consequences for the company. The employees in the department were encouraged to participate in our research project. They were promised that change suggestions would be realised. In the long run the broken promises will probably lead to increasing dissatisfaction among employees in the department.

Management made it clear that they were not interested in doing anything to remove the dissatisfaction that already existed in the company. The employees had to accept the climate as it was for the moment. This management position will however be devastating in the long run.

6.2 CONSEQUENCES FOR GROUP

Many employees stated that they felt that group and individually initiated change in the company was impossible. One person stated, "It feels like walking in syrup". They felt that it was impossible for them to realise desired changes.

The change meetings started by the group constituted a structural and actors change within the department. This change they hoped would lead to subsequent changes in company and departemental structure, technology and tasks. After the failure to achieve the desired subsequent changes the group was left feeling pessimistic, but they still did not want to give up their new feeling of being a group and influencing their *own* work situation. A group where *all* persons made decisions. By participating in the change meetings they had received training in decision-making, taking responsibility, finding new ways of obtaining information etc. This induced them to try another way of change, the internal meeting (see section 6. Group initiated change).

The change meetings were valuable to the material planners in that they found out where the obstacles to changes lay in the company. They found it lay with the AD department, the computer company and especially in the cooperation which did not work well between the AD department and the computer company. The insufficient cooperation between these they perceived as due to routines not having been established yet and to lack of resources in the computer company.

The material planning department's own insufficient cooperation with the purchasing department they decided was partly due to trouble between the division managers and partly to the competitiveness between the departments and the status connected with working in the different departments. As one person in the group stated, "I suppose they must protect their representation accounts". These difficulties were in turn due to the reorganisations.

Other important consequences of the change meetings were that the group obtained a deeper understanding of how certain routines worked and how their work fitted into the company. They had also started to help each other to increase efficiency in their work by sharing individual routines. They also started to discuss prerequisites needed to realise changes; good and continuous contacts with the AD department and the computer company, knowledge about what is going on in the company and a better understanding of what happens with their suggestions, how they are handled and by whom.

The change meetings lead to a permanent change in the group and individual dimensions in that the group decided to continue the departemental meetings where neither the manager nor the researchers would participate. This change will lead to consequences in other dimensions and variables even though it at present is to early to say how.

6.3 CONSEQUENCES FOR INDIVIDUALS

Each of the individuals in the department experienced various positive consequences of the change meetings and the new feeling of being part of a group. Each and everybody started to state their opinions openly and to feel a fully fledged member of the group. The change meetings thus helped to rectify the misalience between the group and individual dimensions that the restructuring and new technology had caused. Even if the individuals were not aware of this misalience they had felt it's consequences. The individuals stated that their job satisfaction and motivation had begun to increase and that these meetings were far to valuable to finish with at the present time.

"Now we have grown up, we can take care of ourselves and do not need you researchers anymore."

"I know we have failed to obtain the changes we need. Now we know that we need support from other groups and managers if we are to succeed."

"It is a good thing to have these meetings, I feel a lot more relaxed in my work and in the group now."

7. DISCUSSION

In our model the consequences of change depends on how it is perceived. We have found that whether change is initiated by only management or by only individuals or groups the change meets with resistance and initiate compensatory or retaliatory changes. If the change is initiated by management the resistance to change occurs at the group and individual level. If the change is initiated by individuals or group at lower levels of the organisation the resistance occurs at management level.

It is our conclusion that change is therefore not only a management problem neither is it only a problem for individuals/groups lower in the organisational structure. "... the prospects for successful institutionalization of a planned change appears to be enhanced by the socialization of the new members, by opportunity for acts of explicit commitment at all hierarchical levels, by the provision of both intrinsic and extrinsic rewards for change, and by making such rewards clearly contingent on the maintenance of the changed behavior" (Goodman et al 1982).

In the moderate run the structural and information technology changes implemented by management in the company where we conducted our study lead to misaliences between certain variables and between the three perspectives. The restructuring within the company caused a turbulent atmosphere. While the profits continued to increase it did not seem likely that the company would rectify this problem. If the company does nothing to rectify these misaliences they will in the long run be in serious trouble concerning their adaptiveness to new societal demands on work quality and their adaptiveness to demands on knowledge in order to remain competitive.

The expressed policy concerning information technology in the company was that all data systems in use should be continuously improved. The material planners had even been promised continual meetings with the computer department, as it was at that time, to discuss which changes they wanted and needed. Until the time of our investigation, a period of three years, not one such meeting had taken place. We concluded that the policy in use was different from the policy expressed.

The implementation of a data system which governed the work of the material planning department made skills in material planning less important in the work. In this way professional knowledge will in the long run diminish and the new material planner will be reduced to a key board pusher. This will have consequences for the company efficiency regarding inventory control and capability to handle unexpected situations in a satisfactory way.

We have shown that management initiated change brought about major consequences due to people´s perceptions of them, at all levels of analysis in the short and the moderate run. The group initiated change did not have any impact on the company in the short run. It had however major impacts on the group itself and individuals in the short run. Group initiated change will however influence the other parts in the moderate and long run. If this was not the case management would not have been so disturbed by the group initiated change process.

We have also shown that neither management initiated nor group initiated change was legitimated in the other parts. Both change processes brought unexpected consequences. The managements by being implemented and the group's by just coming about.

If consequences of change are results of people´s perception of change, then it becomes a common problem for all three parts in an organisation and ***the key to success when change is needed is that it should be based on the active involvement of people from <u>all</u> three dimensions who can participate on equal terms.*** Cooperation between the three dimensions is a necessary requirement for changes that will create positive consequences for the organisation, the groups and the individuals.

References

Bansler J "Systemudvikling teori og historie i skandinavisk perspektiv" (Systems Development, Theory and History in a Scandinavian Perspective), Studentlitteratur, 1987 (in Danish)

Brattgård B, Ingman S "Materialplanering och leveransbevakning" (Material Planning and Delivery Control), research report, Dept of Information and Computer Science and Applied Psychology, University of Lund, LU-ADB-R:89-2, 1989 (in Swedish)

Checkland P "Systems Thinking, Systems Practice", John Wiley & Sons, 1984

Ekvall G "Förnyelse och friktion" (Innovation and Friction), Natur och Kultur, Stockholm, 1988

Eriksson B A "Systemering -från informationsbehov till informationssystem" (Systems Development - from Information Need to Information System), Studentlitteratur, 1986 (in Swedish)

Flensburg P "Systemutveckling med människan i centrum" (Human Centred Systems Development), Studentlitteratur, 1986 (in Swedish)

Frankenhaeuser M, Johansson G, "On the Psychophysiological Consequences of Understimulation and Overstimulation". In Levi L (Ed), Society, stress and disease, Vol IV: Working Life. London: Oxford University Press 1981

Golrang T, Hägerfors A "Människor, datasystem och arbete" (People, Data System and Work), research report, LU-ADB-R:89-?, Dept of Information and Computer Science and Applied Psychology, University of Lund, 1989 (in Swedish)

Goodman P S et al, "Change in Organisations- New perspectives on Theory, Research, and Practice", Jossey-Bass Publishers 1982.

Hedberg B, Elling M, Jönsson S, Köhler H, Mehlmann M, Parmsund M, Werngren C "Kejsarens nya kontor - fallstudier om datoranvändning på kontor (Case Studies on computer Use in Office Work), Liber, 1987 (in Swedish)

Hugoson M, Hesselmark O, Grubbström A "MBI-metoden - En metod för verksamhetsanalys" (the MBI-method - A Method for Analysing Enterprises), Studentlitteratur, 1983 (in Swedish)

Hägerfors A "Design of Information Systems with Dialog in Analysis and Design", in Nissen H-e, Sandström G (eds) "Quality of Work vs Quality of Information Systems", Report of the 9th Scandinavian Research Seminar on Systemeering, University of Lund, 1986

Johansson G, "Psychoneuroendoendocrine correlates of uppaced and paced performance". In G Salvendy and M J Smith (Eds), Machine Pacing and occupational stress. London: Taylor and Francis 1981

Jonson D "Mot en människoorienterad systemsyn - forskningsläge och forskningsbehov" (Towards a Human Centred Systems Approach - State and Need of Research), the Swedish Work Environment Fund and the National Board of Technological Development, 1987 (in Swedish)

Karasek R A, "Job socialization and job strain. The implication of two related psychosocial mechanisms for job design. In B. Gardell and G Johansson (Eds), Working Life. A social science contribution to work reform. Chichester: John Wiley 1981

Kotter J "Organizational Dynamics - Diagnosis and Intervention", Addison-Wesley, 1978

Leavitt H "Applied Organizational Change in Industry: Structural, Technological and Humanistic Approaches", in March J G (ed) "Handbook of Organizations", Chicago, Rand McNally, 1965

Lundberg U and Frankenhaeuser M "Psychophysiological reactions to noise as modified by personal control over noise intensity". Biological Psychology, 6, 51-59 1978.

Maier R F, Verser G C, "Psychology in Industrial Organisations", fifth edition, Houghton Mifflin Company 1982. page 584

Mumford E "Designing Human Systems for New Technology - the ETHICS Method", Manchester: Manchester Business School, 1983

Nissen H-E "On Interpreting services Rendered by Specific Computer Applications", Dissertation, Dept of Information Processing, Royal Institute of Technology, Stockholm, 1976

Nissen H-E, Sandström G, Ekvall G "Ansvars- och medverkansformer i kontinuerlig systemutveckling" (Forms for Responsibility and Participation in Continuous Systems Development), research report, LU-ADB-R:88-3, Dept of Information and Computer Science and Applied Psychology, University of Lund, 1988 (in Swedish)

Ritchie J B, Miles R E, "An Analysis of Quantity and Quality of Participation as Mediating Variables in the Participative Decision Making Process". Personnel Psychology 1970, 23, 347-359

Sandström G "Towards Transparent Data Bases - How to Interpret and Act on Expressions Mediated by Computerized Information Systems", Studentlitteratur, 1985

Wood-Harper T, Antill L, Avison D E "Information Systems Definition: The Multiview Approach", Blackwell Scientific Publications, 1985

Zuckerman M, "Theoretical formulations" In J P Zubek (ed). Sensory deprivation, Fifteen years of research. New York: Appleton Century Crofts, 1969

Tool Support for Cooperative Software Development Tasks in STEPS

Guido Gryczan, Karl Kautz
Technical University of Berlin
Department of Computer Science
Research Group on Software Engineering, Sekr. FR 5-6
Franklinstr. 28/29
D-1000 Berlin 10

Abstract

In this paper we present a tool environment for supporting the cooperative elements of software development tasks. These tools are based on experience we gained in conducting several software engineering projects. Thus, we first outline our view of cooperation in the context of group and project work. In order to classify our tool environment, we discuss related work in the field, which has been a second source for our own ideas. This is followed by a description of our own concepts and an introduction to the tools themselves.

Creativity is doing the right thing in the wrong moment.
(Pelle Ehn, opening the hearing on 'design and creativity', Skagen, Denmark, 1989)

Do the right thing.
(Spike Lee, opening the motion picture 'Do the right thing', New York, USA, 1988)

1 Introduction

The list of quotations about creativity – and right things – could be continued endlessly.

We consider to look at the relationship between cooperation and software development being the right thing in the moment to enhance creativity.

Owing to the complexity and size of the problems to be solved, software development is performed in teams. To make work more efficient, the tasks are divided into subtasks. This leads to group work aspects like cooperation, coordination and communication, which have long been neglected by people engaged in software development. During the last few years, the importance of these aspects has been recognized, and increasing interest is evident, especially in the field of computer supported cooperative work. Several approaches to improving cooperation in teams have been adopted, ranging from linguistically and sociologically based to pragmatic ones. However, there is a wide range of different interpretations of these notions in the field of computer science (see for example, /CSCW86/, /CSCW88/, /Greif88/).

While conducting software engineering projects for several years, we recognized that many of the problems arising during the development process were associated more with the cooperative aspects of the work than with technical ones. This motivated us to turn our attention to this field, and resulted in a proposal for a cooperation support tool environment we will describe here.

This paper, thus, will proceed as follows :

In section 2 we present our own experiences in project work and the kind of cooperation found there. This is brought into relationship with what others say about cooperation and serves as a first basis for our tools.

Section 3, then, gives an overview about existing tools supporting different concerns of cooperation. This has been a second source for our own ideas.

In section 4 we concretize our view of project work embedded in an approach taking software development as design /Floyd89.2/, which is the third basis for the concepts of the tools.

Having described the conceptual background so far, in section 5, we outline the technical basis for the tool environment.

In sections 6 we describe, what we consider to be the material designers work with, and in section 7 we introduce the tools themselves, which we consider to support the cooperative work with these materials.

2 Cooperation in Software Development - Our experience

When we talk about cooperation in the context of software development, we are primarily referring to our experiences with projects involving cooperation between developers in the university and research laboratory context. In this context, external pressures such as political and economic issues influencing the conditions of work, are of comparative insignificance.

The project groups, we were conducting, worked on the basis of the methodological approach STEPS /Floyd87/, /Reisin88/, which incorporates a process-oriented view of software development, a human-centered notion of quality, and a cyclic project model.

The groups have little internal hierarchy and are relatively autonomous. The problems and tasks, the project members work on, cannot be defined completely. In addition to this kind of task uncertainty, there is no sharp division of the work tasks as everybody is involved in planning, documenting, designing and implementing. And there is also great flexibility in the distribution of the tasks. This is a context where cooperative work can evolve /Andersen87/, /Sørgaard88/, /Dzida83/.

But as human beings are involved, we have to distinguish a task and an interpersonal relationship level, which influence each other /Kraut86/. There is a social context, which is characterized by aspects such as belonging to a group and personal trust.

Successful group work to a large extent depends on a shared goal of the project members /Ciborra88/. This means, establishing and maintaining a common perspective and vision, about how the project should proceed and what result should be achieved. This is what other authors call a group culture, in which groups share assumptions and work on the basis of commitments /Goodman87/. Acquiring and maintaining a common understanding of the product, therefore, is an important and permanent part of group work.

All this is largely performed through informal communication on the basis of the information and material the project members share. Discussions, negotiations and conflicts are taken as a fruitful source for evaluating ideas, taking decisions and achieving a consensus about the systems to be developed. They serve to resolve disagreement and to clear up misunderstandings.

Cooperative work, then, is more for us than working individually on a number of tasks resulting in a common product; more than coordinating people and their tasks; and more than merely exchanging information.

In contrast, traditional software development models and concepts see group work as a necessary burden. Their main purpose is to coordinate individuals working on separate tasks. The groups are centralized. Strong hierarchies and fixed rules are used to reduce communication and information exchange, and to control the development process /Buxton70/, /Greenbaum88/, /Pasch89/. Such concepts are based on fixed phases and fixed division of work /Baker72/, /Budde89/, /Gryczan88/ within a product-oriented view of software development. This is not the kind of work and organization we have in mind when talking about cooperative software development. In a context like this, real group work can scarcely evolve.

When we consider the way people gather and use information in our project work, following the STEPS approach with regard to the organizational structure, we observe that the groups are not static, that they change due to both the relationships of the group members and the evolving tasks. This does not mean that there is no organizational structure, with different roles, tasks, rules, duties and rights, but it is based more on group consensus and change during a project.

The teams are divided into component groups, component groups into even smaller groups, down to the level of individual project members, according to the structure of overall and partial tasks. Members of different groups form bridging groups to exchange the results of their work. The whole team, but also every component group, is involved in a permant process of intertwined organizational and substantial work.

Cooperation takes place in communicating information, either spoken or written. On every group level, face-to-face meetings, moderated by changing project members, serve for creating and evaluating ideas and making common decisions. The meetings are supported through, and the results are stored in the form of, individual and common notes and official public proceedings. These also constitute the basis for the documents representing the 'final' results of the projects.

Even tasks that seemed to be mainly performed by individuals - at least over a certain period - such as producing and editing system descriptions and code, have cooperative aspects which influence the work of others involved in the same, or a related, task.

Documentation and written information is an important source of communication and cooperation between teams and individuals, especially in the case of spatial or temporal absence of project members whose information is needed to continue a task. These experiences have been recently confirmed by /Curtis88/, who conducted an extensive field study concerning software engineering projects.

Owing to the effort involved - the natural writing expenses additionally augmented by inadequate tools and unnecessary bureaucracy - much of the information and many of the decisions are not documented. This makes the documents inconsistent and misinterpretable. Hence lost information makes cooperation more difficult.

Since we believe that information exchange and documentation as parts of cooperative behaviour are extremely unstructured processes, they cannot been enforced by directions. When thinking of supporting these processes, we have tools in mind for facilitating provision and storage of the necessary information, which in addition to the existing communication channels, may support some aspects of cooperative work.

3 Cooperation Support Tools - An overview

On this basis and with our own experiences as a background, we analyzed existing tools with respect to their suitability in supporting cooperative work processes in software development. Most of them turned out to deal with more general characteristics of cooperation. To get a structured overview, we, therefore, took the main concern of the systems as a means for categorization. A discussion of the different theoretical and/or technical basis, which also were part of our investigations, is far beyond the scope of this paper. Thus, on a more general level, we have distinguished systems for

- meeting augmentation

- people and work task coordination
- general information exchange
- joint document writing and editing

(for other classifications, see /Wilson88/, /Kraemer86/ /Opper88/). Additionally, some systems combine several of these aspects.

Meeting augmentation systems like Cognoter /Foster86/ serve as a means for supporting several kinds of meetings /Begeman86/. They provide every participant with a monitor, and use an additional large-scale screen to visualize ideas textually and graphically on the basis of the 'chalkboard' metaphor /Stefik87/. These tools are used to support the creation and evaluation of ideas by making them transparent. Specific rules are applied to conduct the indiviual sessions. The computer serves as a shared, extended group memory to resume discussions, to revisit previous argumentation, and to retrieve information generated at previous meetings. However, no means are provided for communicating and coordinating work outside the sessions.

The central aspect of coordination systems like Coordinator /Winograd86/ and Chaos /De Cindio86/ is the distribution of tasks. To achieve commitments, typed messages such as explicit request, acceptance, offer and completion are provided to direct conversations according to predefined rules. To keep track of their fulfilment, information about unanswered messages, incomplete conversations and unfinished tasks can be retrieved by both the sender and the receiver of messages on the basis of information pools. As they are based on a very rigid interpretation of speech acts /Searle71/, they work in a highly prescriptive way.

Information exchange systems like electronic mail, bulletin boards and conferencing systems are based on traditional forms of communication /Hiltz85/. They support general person-to-person and/or person-to-group communication. Information is sent as a written text message. Messages can contain all kinds of texts, for example, also program sources. They can be transformed to files, this feature allows messages to be used by other tools. Messages can be partially structured, providing information about who has written what to whom. As they can be stored, it is possible retrieve information. This is restricted to an individual level as there is no explicit shared pool for commonly organized, group-relevant information. Thus, it is difficult to achieve feedback about which member of a group has received a certain information and has read it. As structuring information is bound to normal file operations, most of these systems provide inadequate means for easy use in both organizational and substantial work.

Document-editing systems support a certain aspect of cooperative work. They enable text documents to be written by more than one author. Systems, which claim to support this in a cooperative way, like NoteCards /Trigg86/, or Quilt /Leland88/, are based on a special model of documents. Text, there, before being linearized, is taking as a non-linear, connected structure of text fragments. This model is implemented in the so-called hypertext systems (see section 5), where non-linear text is represented on small note cards in a direct manipulation interface environment. The cards can be connected by links to networks of cards /Conklin87/. This allows non-linear structuring of ideas, drafts and shared text to be organized in different ways before being committed to normal linear documents. Cards and already exisiting document parts are stored in a shared data base.

There are various ways of dealing with the access rights of shared text. In NoteCards, this must be negotiated outside the system, and the conventions must be controlled by the authors without support; in Quilt, different roles can be defined giving different access rights to different people. In general, working on a shared text is supported by special features recording information pertaining to changes. Overall coordination and organization is not supported explictly, although it is possible to use special cards to do so.

Considering systems that combine several ideas /Dzida83/, /Kedzierski88/, /Conklin88/, the Information-Object Lens /Malone86/, /Lai88/ seems to be the most advanced. It connects concepts of communication theory with technical features of electronic mail and hypertext systems in order to support written communication. The system is based on the idea of using semi-structured objects represented by cards for exchanging and providing information. The benefits of semi-structured cards are explained by Malone as follows:

- The structured part of the cards allows information to be processed by a software system, providing the user with means of individually filtering relevant information.
- Sending standard information is supported by using the structured part.
- The structured part facilitates the recognition of the main information of messages.
- The unstructured part allows the free composition of messages.

The system allows exchange of typed messages, which can be archived in a shared information pool. Message cards provide fixed fields for standard information about the sender, the sending date and the type and subject of a message, and a free format field for formulating the message. The predefined messages types can be used to make the contents of a message explicit. The system makes suggestions for reply message types, but does not require any special answer type. This concept of semi-structured cards is extented to other information objects useful in the context of project work. Roles of persons for example can be defined by using semi-structured templates. Additionally, objects containing all kinds of possible information can be organized and connected by links known from hypertext systems. The Object Lens, like the other tools of this group mentioned above, supports many aspects of written communication, but does not emphasize on supporting people working on coherent and interdependent tasks.

To summarize our analysis, each of the systems presented here, covers some aspects of cooperative work, which may be combined and applied usefully to support cooperative software development in particular. Thus, before explaining, which of the concepts we adopted for our tool environment, we now concretize our view of cooperative software development.

4 Our View on the Design Process

Following /Floyd89.2/ we regard software development as a *design process*. According to this view, one of the primary goals is to achieve a common perspective of the desired functionality of the system. The decisions made during the design process must reflect the different perspectives on the system, i.e. those of the management, the developers and the users. The usability of these decisions can only be assessed by experience and judgement. As a consequence, feedback is a crucial part of this approach.

To attain this goal, the environment has to support both, the process-oriented and the product-oriented aspects of a project. Besides those characteristics already described, from our point of view, the relationships between roles, taken by project members, tasks and products constitute and influence a common working milieu in a project.

The inter-dependencies between them are present within each development step of a project. In the following, we will refer to /Floyd89.1/ by naming prerequisites for dialogical design. We try to define how these requirements can be realized in a tool-based environment, emphasizing the establishment of a project, this being the crucial period for setting up a common working milieu. To establish a project, the following tasks should be supported:

- Determining the project goal. The project goal will change as the project evolves. Change, in this respect, means stepwise concretization with the increasing penetration of the application context. The environment has to support such a redefining of the project goal.

- Making transparent the way in which responsibility for tasks is handled. That means, the project members define conventions as to how tasks and responsibilities for these tasks are to be managed. The environment should support a variety of mechanisms for managing the tasks evolving during the project.

- Assuming roles by project members. Each member of the project will assume one or more roles at the same time depending on different tasks. These roles also can change during the project. This means that each project member will be responsible for certain tasks. The environment should support any needs for changes in responsibilities.

- Defining *reference lines.* In keeping with /Andersen86/ a reference line is defined by tasks that should be fulfilled at a specific time. If this date is reached, the state of tasks is revisited and according to this state a new reference line is defined. All actual tasks should be transparent to all project members. We do not consider the way decisions are taken that lead to definition of a reference line. What we want to achieve is, that the results of the decision-making process are made transparent.

4.1 Tasks and Functional Roles

The adaptation of tasks is usually tied to a specific role or to a specific person. The concept of *functional roles* can serve as a basis for the description of tasks in a development project. A functional role is characterized by

- the integration of tasks

- an assignment of persons and tasks

- the responsibility of a person for tasks

Take, for example, the implementation of a particular module. This task will be fulfilled by a group of people. It is desirable, then, for other members of the project to communicate with them by calling them, say, I/O group. Within this group, requests can be forwarded to specific persons. Roles are determined by tasks, and project members can assume one or more roles within the project. On the other hand, it is not necessary for a role, for example, that of the project moderator, to be assumed by a specific person. That means, the definition of a role is not automatically bound to a single person. Furthermore, the responsibility for a functional role can be changed during the project.

From our experience in software projects, we distinguish the following roles:

- Project Moderator: The moderator is responsible for the administration of work tasks and also for the establishing of reference lines.
- Designer: Designers are responsible for creating and shaping the product. Accordingly, designers have far-reaching rights for making annotations and comments, not only to their own work, but to the work of others as well.
- Programmer: Programmers are responsible for performing programming tasks according to the consensus reached as the result of design.
- Editor: Editors are responsible for organizing the development and for layouting documents to be read by users or management, i.e. to control the style of documents.
- Tester: Testers are responsible for the validation of software according to rules set by the project.

These roles are not the only possible ones, but more important they are not intended to lead to a hierarchical project model, as known for example from the chief programmer team /Baker72/.

The functional roles can be related to different useful ways of viewing the project. In the following we present three different views, each characterized by an interrogative word.

- Who: The role view gives an overview of tasks and the state of results and products that are assigned to a specific role.
- What: The document view supports the handling of named documents representing informal information, results and products in manageable sizes.
- When: The reference view shows by which date certain tasks within roles have to be completed; it serves as a basis for a project-global coordination of work tasks.

It should be possible for each member of the project to assume every role defined in the project, and to adopt any specific view appropriate to actual needs.

4.2 Documenting in Software Development

During the development process, several kinds of documents are generated. By the term *document*, we mean all kinds of written material. Therefore, we distinguish different classes of documents.

Product Documents are documents, that serve as parts of contracts and as parts of the product to be delivered. This means in particular that requirements and functional specificatons are part of this kind of documents. Glossaries as results of a process of finding a common language and defining shared concepts should also be product documents. Another kind of product documents are User Documents which are documents to be read by the later user explaining how to use the system.

On the other hand Process Documents are documents dealing with cooperation between project members. These documents record decisions that shape the structure of the system developed. Examples of documents of this sort are individual, subgroup and project-global diaries and notebooks. Protocols reflect the decision making process in a project.

There are many, partly hierarchical, relationships between documents that are not usually supported. For example, the implementation documentation of a piece of code is related to a part of the design documentation and to a test case documentation for this code; and may be related to a description of tools necessary to integrate this code into an entire programm. Additionally, as already mentioned, there may and should be several documents describing the decisions which led to implement this piece of code.

5 Our technical Basis

It is obvious that dependencies between documents cannot be represented in a hierarchical way. Hence, an approach dealing with non-linear structures, like Hypertext, can serve as a basis for the kind of problems occurring with respect to documentation, i.e. the mutual dependencies between the documents. Even more important, Hypertext can serve as a basis for representing the processes occurring during the development of documents. In /Trigg88/, the author states: *hypertext systems have the potential to represent both the work and the accompanying descriptions and meta-discussions.*

For our system we reinterpret the concept of non-linearly connected units to apply to objects of meaning within the software development process. A restriction is our confinement to textual objects, in contrast to Hypertext-systems, where any kind of object can be related to another. Additionally, we restrict link-operations to words as the smallest units, so that we only have to deal with two classes of links. There exist different kinds of links between documents, document cards and words. Take, as an example, the word *train* in the context of a systems development. It should be clear that there exist different definitions of what a train actually is, according to different perspectives. Now, if there is an easy way to look up in which way the term is used, for example by a define-link, misunderstandings of this kind can be minimized.

The primary reason of our confinement to textual objects is of technical nature. In future versions, we will integrate graphical objects, like diagrams to specify the module structure of the system.

structure of the system.

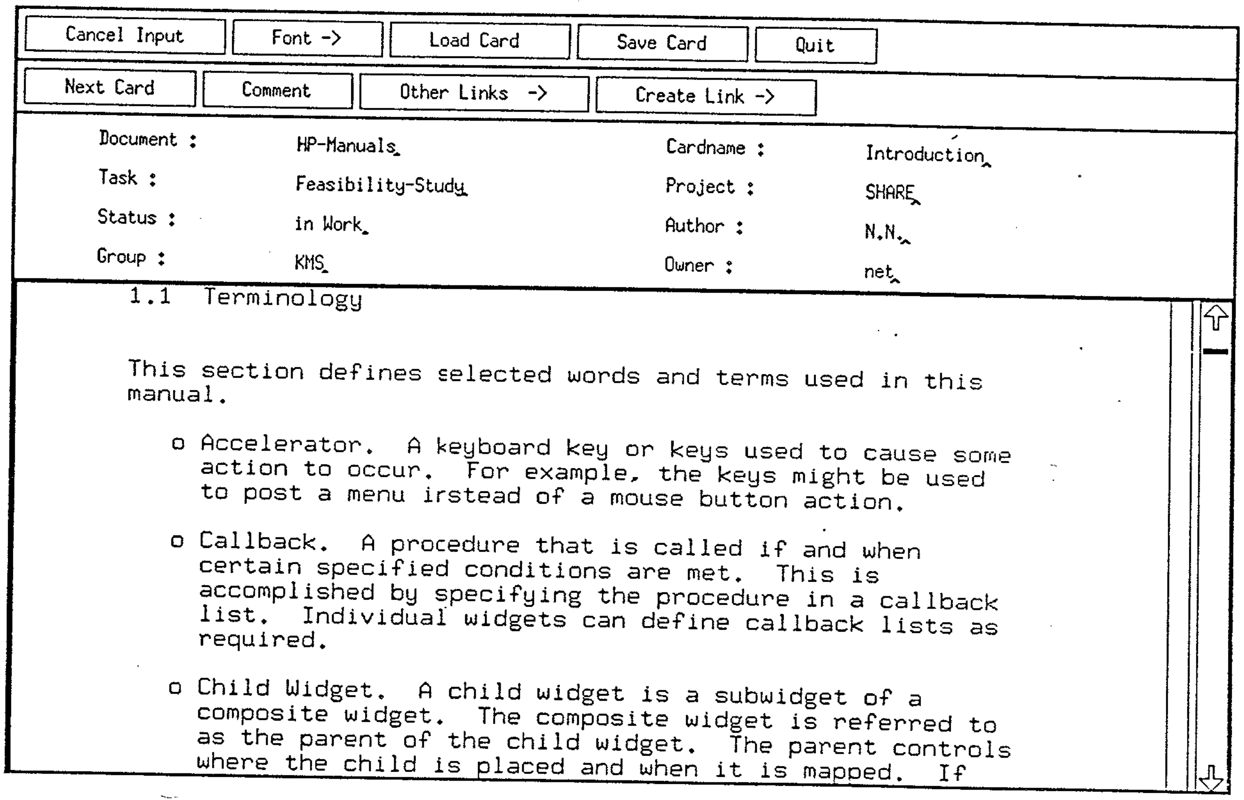
To sum up, we do not intend to develop a new Hypertext-system, but a system where

- the *product* and the accompanying *processes* can be represented in a single medium, and
- the dynamic and the static aspects of the *product* and the accompanying *processes* can be described.

The latter point refers in particular to *programs* and the *documentation processes* necessary to understand and maintain these programs.

6 The working Material

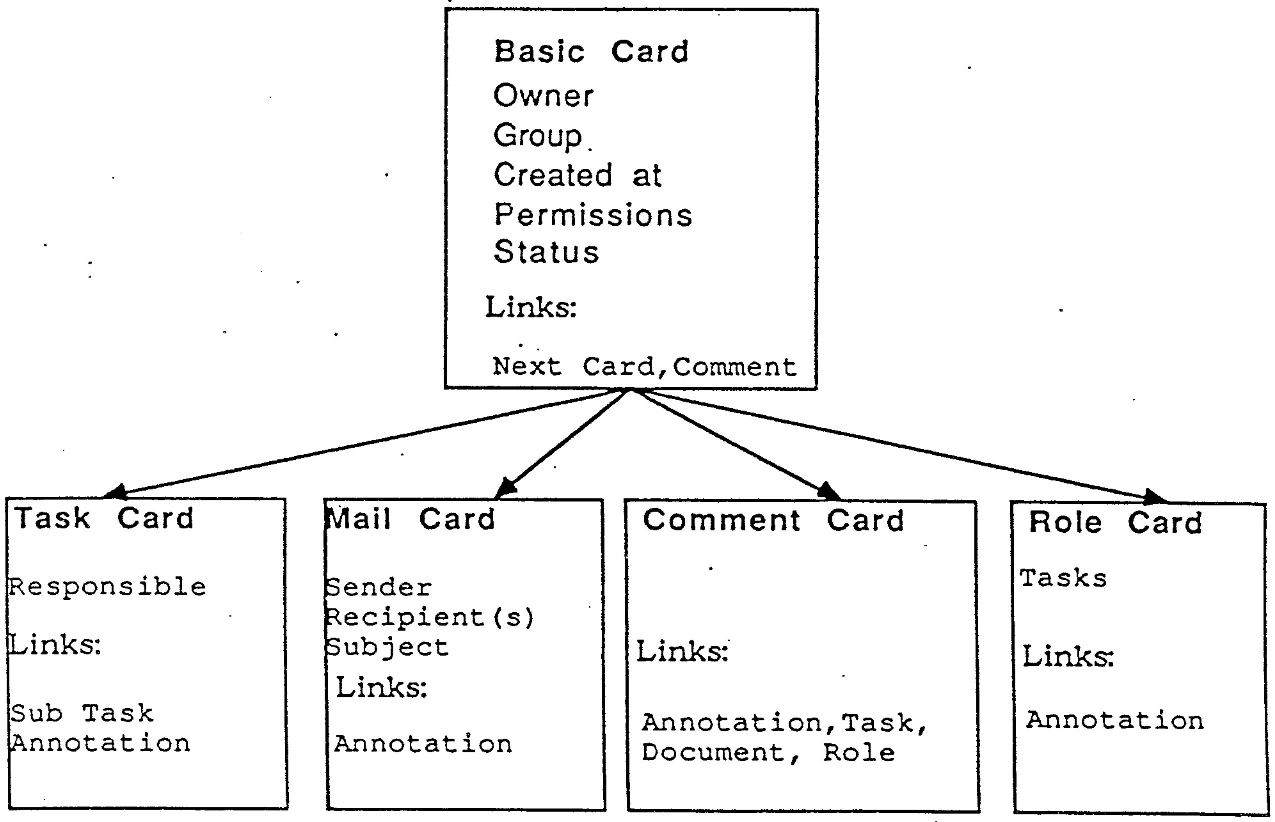
We consider information in documents to be the basic material within a project. In the preceeding section, we have outlined that Hypertext can serve as a basis for our technical environment. According to the Hypertext idea, our objects of concern, the documents, are represented by a net of cards. Cards are the main objects for recording information, they can be divided up into a structured and an unstructured part. The relation of a card to other cards is determined by its type and its connection to other cards, i.e. its being connected in a logical or chronological order. The figure below shows the basic card.



Document :	HP-Manuals	Cardname :	Introduction
Task :	Feasibility-Study	Project :	SHARE
Status :	in Work	Author :	N.N.
Group :	KMS	Owner :	net

```
1.1  Terminology

This section defines selected words and terms used in this
manual.

   o Accelerator.  A keyboard key or keys used to cause some
     action to occur.  For example, the keys might be used
     to post a menu irstead of a mouse button action.

   o Callback.  A procedure that is called if and when
     certain specified conditions are met.  This is
     accomplished by specifying the procedure in a callback
     list.  Individual widgets can define callback lists as
     required.

   o Child Widget.  A child widget is a subwidget of a
     composite widget.  The composite widget is referred to
     as the parent of the child widget.  The parent controls
     where the child is placed and when it is mapped.  If
```

On each card, operations known from windowing systems can be performed such as moving or resizing. Operations known from Hypertext-systems are possible, such as the creation and deletion of cards. The structured part contains context-related fields, like the author field, and commands related to the handling of the cards like 'quit'. The basic card shown above contains already a number of features useful in creating documents and communicating about them. But, for specific tasks, one may need specialized cards. We therefore propose and provide a class (hierarchy) of cards to perform specific tasks. The figure below shows classes of cards useful in different situations. The message card for example can be used for explicit written communication within the project.



7 The supporting Tools

Lost In Hyperspace is a danger known from usage of Hypertext systems. To avoid this danger, we are developing tools to generate specific views on the project material. All tools presented here, work according to the same concept. After a call of one of the tools, the actual material, cards or complete documents, is presented in an iconified manner, with links visible between them, if there are any. To work on a specific card, it is de-iconified. It is possible to work on more than one cards at a time, and to use more than one tool simultaneously. The arising problem of parallel work on the same cards is solved by a simple locking mechanism based on semaphors, that prevents simultaneous write access to objects.

7.1 Share

Share is a tool intended to allow specific views on the project material, i.e. the document view and the person/role view as mentioned above.

Like the concept of the cards, Share is on the interface, being divided into a structured and an unstructured part. In the structured part, it is possible to specify a view. By default, all cards are visible.

In the *document view*, named documents are visible through icons representing them. These documents comprise requirements analysis and the like. To work on these documents, they are de-iconified, and the cards structure building them is visible.

In the *person/role view*, all documents belonging to a specific person/role are visible. Accordingly, this view gives an even more restricted view on the material. This view is supposed to be the usual working view while developing software or documentation.

Additionally, we are integrating a rule base into the tool. With this rule base, it will be possible to have a more specific focus on the material of concern.

7.2 Reference

Reference is intended to support the administration of reference lines to keep track of and to coordinate responsibilities, tasks and results. Reference can be used in two ways. The usage of this tool similar to the use of Share.

It is possible to obtain an overview of the existing tasks and the current state of documents. Tasks are represented graphically on a time chart, thus each project member can see, who should have performed a specific task upto a particular time.

The second use of Reference is that of a reminder tool, where each project member can remind her/himself of tasks, which are tied to specific reference lines.

7.3 Diary

Diary should support the keeping of project diaries. Decisions made at meetings, or anything else of particular interest to the diary writer, can be kept in a diary, private to a role or person. Private diaries can be kept unvisible to other project members.

The tool also supports the keeping of a project-global diary. The maintenance of this diary should be done by the project moderator. The project-global diary is of significant importance to keep the history of the project.

7.4 Inform - Newsboard

The tools Inform and Newsboard are closely connected to the keeping of the Diary. With Newsboard an electronic notice board is provided. The use of this tool is highly dependant on the working milieu of the project. If the use of this tool is not sufficient for the dissemination of project relevant information, the use of Inform is indicated.

Inform makes use of the above defined message card. It can be used as a project specific mail-tool as it supports the sending of messages to specific roles.

For both tools it is planned to implement a rule-based component like the one for Share.

7.5 Glossary

With the tool Glossary, keywords of a project can be maintained. Its main purpose is to reach a common understanding of important notions. For this reason, notions and their use in the project are stored, including links to cards, where these notions occur. Each item in the glossary is explained by its structure, attributes and by a short description.

While working with a card, it is possible to check whether a term is already stored in the Glossary. As with other tools, it is possible to make comments. Here, it is up to the project in which way changes are allowed to make.

7.6 Future Efforts

The tools presented above build a first step to an usable environment supporting the cooperation and communication within development teams. According to our view on software development the shape of the environment is subject to change as our experiences with it are growing.

8 Summary

In this paper, we have looked at aspects of cooperation and communication in software development teams. We, then, went on to look at existing concepts and tools for supporting cooperative development tasks. Finally, we presented our own concept and environment for supporting the collaboration in development projects focusing on information exchange and storage for both the group and the individual members to ease the establishment and maintenance of group culture, tradition and knowledge by keeping track of the decisions concerning the division of labour and the resulting (intermediate) products. Many problems still remain to be solved. In particular, we have not yet integrated a prototyping environment. This appears to be a crucial point. Another task will be the integration of task-specific tools, for example a tool for requirements analysis. We have not yet defined a strategy concerning tools outside our environment.

Finally, to state it again, we want our environment to be understood as an offer for people involved in software development. They should have a maximum of freedom to choose any tool, they are used to.

Literature

/Andersen86/ Andersen, N. E., Kensing, F., Lassen, M., Ludin, J., Mathiassen, L., Munk-Madsen, A., Sørgaard, P., Professionel Systemudvikling, Kopenhagen, 1986

/Andersen87/ Andersen, P. B., et al., Research Programme on Computer Support in Cooperative Design and Communication, Aarhus University, Denmark, 1987

/Baker72/ Baker, F. T., Chief Programmer Team of Production Programming, IBM Systems Journal, No. 1, 1972

/Begeman86/ Begeman, M., Cook, P., Ellis, C., Graf, M., Rein, G., Smith T., Project NICK: Meetings Augmentation and Analysis, in Proceedings of the Conference on Computer-Supported Cooperative Work, pp. 1-6, Austin, Texas, 1986

/Budde89/ Budde, R., Kautz, K., Kuhlenkamp, K., Zuellighoven, H., Prototyping - An Approach to Evolutionary Systems Development - Springer-Verlag, Berlin, Heidelberg, New York, Tokyo, to be published 1990, 1989

/Buxton70/ Buxton, J. N., Randell , B. (eds.), Software Engineering Techniques, NATO Sciences Committee, Garmisch, April 1969

/CSCW86/ Proceedings of the Conference on Computer-Supported Cooperative Work, MCC Software Technology Program, Austin, Texas, December 1986

/CSCW88/ Proceedings of the 2nd Conference on Computer-Supported Cooperative work, Portland, Oregon, 1988

/Ciborra88/ Ciborra, C. C., Olson, M., H., Encountering Electronic Work Groups: A Transaction Costs Perspective, in Proceedings of the 2nd Conference on Computer-Supported Cooperative work, pp. 94-101, Portland, Oregon, 1988

/Conklin87/ Conklin, J., Hypertext: An Introduction and Survey, in Computer, IEEE, The Computer Society, September 1987, pp. 17-40, 1987

/Conklin88/ Conklin, J., Begeman, M. L., gIBIS: A Hypertext Tool for Exploratory Policy Discussion, in Proceedings of the 2nd Conference on Computer-Supported Cooperative work, pp. 140-152, Portland, Oregon, 1988

/Curtis88/ Curtis, B., Krasner, H., Iscoe, N., A Field Study of the Software Design Process for Large Systems, in Communications of the ACM, Vol. 31, No. 11, pp. 1268-1287, November 1988

/De Cindio86/ DeCindio, F., De Michaelis, G., Simone, C., Vassallo, R., Zanaboni, A. M., Chaos as coordination technology, in Proceedings of the Conference on Computer-Supported Cooperative Work, pp. 325-342, Austin, Texas, 1986

/Dzida83/ Dzida, W., Spittel, A., Sylla, K.-H., Einsatz eines "Message System" bei der Software-Entwicklung - Ein Erfahrungsbericht, In: Schauer, Helmut und Tauber, Michael J., Psychologie des Programmierens, R. Oldenbourg Verlag, Wien, Muenchen, 1983.

/Floyd87/ Floyd, C., STEPS- eine Orientierung der Software Technik auf sozialverträgliche Technikgestaltung, in Informatik Forum, 2.Jg., Heft 2, pp. 40-45, 1987

/Floyd89.1/ Floyd, C., Softwareentwicklung als Realitätskonstruktion, to be published in The Proceedings of the Conference Software-Entwicklung - Konzepte, Erfahrungen, Perspektiven, held in Marburg/Lahn from 21-23 June, 1989

/Floyd89.2/ Floyd, C., Reisin, F.-M., Schmidt, G., STEPS to Software Development with Users, to be published in The Proceddings of ESEC 1989, to be held in Warwick, England, from 11-15 September, 1989

/Foster86/ Foster, G., Stefik, M., Cognoter, Theory and Practice of a Colab-orative Tool, in Proceedings of the Conference on Computer-Supported Cooperative Work, pp. 7-15, Austin, Texas, 1986

/Goodman87/ Goodman, G. O., Abel, M. J., Communication and Collaboration: Facilitating Cooperative Work through Communication, in Office Technology and People, Vol.3, No.2, pp. 129-145, Elsevier Sc. Pub., Amsterdem, 1987

/Greenbaum88/ Greenbaum, J., In Search of Cooperation - An Historical Analysis of Work Organization and Mangement Strategies, in Proceedings of the 2nd Conference on Computer-Supported Cooperative work, pp. 102-114, Portland, Oregon, 1988

/Greif88/ Greif, I. (ed.), Computer-Supported Cooperative Work: A book of Readings, M. Kaufmann Pub., Inc., San Mateo, California, 1988

/Gryczan88/ Gryczan, G. , Kautz, K., Research Activities on Evolutionary System Development at the Technical University of Berlin (West) - An Evolutionary Process, presented at the Polish-Scandinavian Seminar on Current Trends in Information Systems Development Methodologies, Paraszyno, Poland, June, 1988

/Hiltz85/ Hiltz, S. R., Turoff, M., Structuring computer-mediated communication systems to avoid information overload, Communication of the the ACM, 28, 7, pp. 680-689, July 1985

/Kedzierski88/ Kedzierski, B. I., Communication and Management Support In System Development Environments, in Computer-Supported Cooperative Work: A book of Readings, Greif, I. (ed.), pp. 253-268, M. Kaufmann Pub., Inc., San Mateo, California, 1988

/Kraemer86/ Kraemer, K., King, J., Computer-Based Systems for Group Decision Support: Status of Use and Problems in Development, in Proceedings of the Conference on Computer-Supported Coperative Work, pp. 353-375, Austin, Texas, 1986

/Kraut86/ Kraut, R., Galegher, J., Egido, C., Relationships and Tasks in Scientific Research Collaboration, in Proceedings of the Conference on Computer-Supported Coperative Work, pp. 229-245, Austin, Texas, 1986

/Lai88/ Lai, K.-Y., Malone, T. M., Object Lens: A "Spreadsheet" for cooperative Work, in Proceedings of the 2nd Conference on Computer-Supported Cooperative work, pp. 115-124, Portland, Oregon, 1988

/Leland88/ Leland, M. D. P., Fish, R. S., Kraut, R. E., Collaborative Document Production Using Quilt, in Proceedings of the 2nd Conference on Computer-Supported Cooperative work, pp. 206-215, Portland, Oregon, 1988

/Malone86/ Malone, T. M., Grant, K. R., Lai, K.-Y., Rao, R., Rosenblitt, D., Semistructured Messages are Surprisingly Useful for Computer-Supported Coordination, in Computer-Supported Cooperative Work: A book of Readings, Greif, I. (ed.), pp. 311-331, M. Kaufmann Pub., Inc., San Mateo, California, 1988

/Opper88/ Opper, S., A Groupware Toolbox, in Byte Magazine, Vol. 13, No. 13, McGraw-Hill Pub., pp. 275-282, 1988

/Pasch89/ Pasch, J., Mehr Selbstorganisation in Softwareentwicklungsprojekten, Workshopbeitrag auf der Fachtagung Software-Entwicklung; Konzepte, Erfahrungen, Perspektiven, Marburg/Lahn 21.-23. Juni 1989

/Reisin88/ Reisin, F.-M., Schmidt, G., STEPS - Ein Ansatz zur evolutionären Systementwicklung, computer magazin, Vol. 17, No.7/8, 1988

/Searle71/ Searle, J. R., Sprechakte - Ein sprachphilosophischer Essay, Suhrkamp Taschenbuch Wissenschaft, Frankfurt am Main, 1983, Erstausgabe, 1971

/Sørgaard88/ Søgaard, P., A Discussion of Computer Supported Cooperative Work, Ph. D. thesis, Aarhus University, Denmark, 1988

/Stefik87/ Stefik, M., Foster, G., Bobrow, D. G., Kahn, K., Lanning, S., Suchman, L., Beyond the Chalkboard: Computer support for Collaboration and Problem Solving in Meetings, Communications of the ACM, Vol. 30, No. 1, pp. 32-47, January 1987

/Trigg86/ Trigg, R. H., Suchman, L. A., Supporting Collabaration, in NoteCards, in Proceedings of the Conference on Computer-Supported Cooperative Work, Austin, Texas, 1986

/Trigg88/ Guided Tools and Tabletops: Tools for Communicating in a Hypertext Environment; Proceedings of the 2nd Conference on Computer-Supported Cooperative work, pp. 216-226, Portland, Oregon, 1988

/Wilson88/ Wilson, P., Key Research in Computer-Supported Cooperative Work (CSCW), Proceedings of the European Teleinformatic Conference - Euteco '88, Speth, R. (ed.), Vienna, Austria, pp. 211-226, North Holland, Amsterdam, New York, Oxford, Tokyo, 1988

/Winograd86/ Winograd, T., A Language Perspective on the Design of Cooperative Work, in Proceeding of the Conference on Computer-Supported Coperative Work, pp. 203-220, Austin, Texas, 1986

EXTENDING THE BOUNDARIES OF PROTOTYPING - TOWARDS COOPERATIVE PROTOTYPING

*Kaj Grønbæk**
Computer Science Department,
University of Aarhus, Denmark.

Abstract

The term "prototyping" has become a buzzword in both research and practice of system development in recent years due to a number of agreed advantages of prototyping techniques compared to traditional system specification techniques. In this paper the "state of the art" of prototyping approaches is critically assessed with regard to how user involvement and creativity is treated. It is concluded that the mainstream of prototyping approaches currently applied and advocated in literature are poor with regard to support user involvement and creativity in systems design.

This conclusion leads to the introduction of a *cooperative prototyping* approach that better involve users creatively in system design. The aim of this approach is to build tailored computer systems that enable users to do their work with a minimum of breakdown situations caused by the computer system. The technique is to establish a design process based on cooperation between users and designers through a close coupling between early *evaluation* of ideas in an envisioned use situation with prototypes and *design* through rapid modification of the prototypes. Computer-based tools to support rapid development of prototypes that are *directly manipulable*, are emerging. Examples of such tools are given, and it is discussed how they can support cooperative prototyping within certain use domains.

1 Introduction

Recently literature on prototyping "methods" and techniques has emerged, and it is slowly replacing the literature on structured methods of the 70s. This is illustrated by book titles such as "The Prototyping Methodology" (Lantz 1986), "Application Prototyping - A requirements definition strategy for the 80s" (Boar 1984), and "Software Prototyping, Formal Methods and VDM" (Hekmatpour & Ince 1988).

With my background in a system development tradition[1] taking the users practice as the starting point for new developments, it is natural to ask the question: How does this methodological change of focus improve the users' opportunities to be involved and get influence on system development in their organization ? To study this question, I have formerly been engaged in empirical studies of the use of prototyping techniques in IS development projects, described in (Grønbæk 1988). These studies showed that even though the developers used prototyping approaches similar to recommendations given in the literature mentioned, the users were only brought in to evaluate horizontal prototypes based on demonstrations. These activities did not result in a significantly better system designs than earlier description methods did - a number of new requirements were raised when the users experienced the new system during installation activities. These observations gave motivation to investigate how prototyping techniques and tools can be improved to enable a more active and even creative involvement of users in system development. Elements of this investigation has been carried out by studying the use of a so-called *cooperative prototyping* approach in particular use domains, an example is described in (Bødker & Grønbæk 1989). In this paper I will support the empirical studies with a critique of the current prototyping approaches as described in literature, and I will introduce the ideas of cooperative prototyping supported with a brief survey of the technological possibilities for development of tools to support this prototyping approach.

The structure of the paper is as follows: In section 2 I give a critical assessment of how current prototyping techniques and methods, described in literature, support user involvement and creativity in system design. A number of factors that contribute to increasing user demands on better quality computer systems in the future are also briefly discussed to emphasize the need for user involvement. In section 3 the needs for extending the boundaries of prototyping towards system design with users is emphasized, and a *cooperative prototyping* approach that support user involvement and creativity, is proposed. The potential sources of choosing and developing computer support for cooperative prototyping are discussed in section 4, through a brief survey of examples on how various *direct manipulation tools* support rapid (possibly quick-and-dirty) development and modification of prototypes within particular use domains.

2 State of the art of prototyping

In this section I will give a critical assessment of current prototyping approaches from the literature with regard to how they support active and creative user involvement in the design process. This critical assessment is based on the assumption that active user involvement in design is the key to anticipate the problems traditionally seen when a new computer system is installed in a work-place. A number of authors, e.g. (Floyd 1984, Hekmatpour & Ince 1988, Kammersgaard 1984), have listed categorisations of prototyping techniques at various levels of abstraction and with various perspectives. However, for my discussion in this paper I will use yet another rough categorisation of approaches to prototyping, to deal with the mainstreams of prototyping approaches, as they are described in current literature. The literature and thus the approaches roughly fall into three categories that can be denoted: 1) "The Prototype becomes the system" approaches, 2) Executable specification approaches, 3) Exploratory approaches. For the discussion of the individual approaches I am using a framework introduced in (Mathiassen 1981) for describing and assessing system development methods. In this framework distinctions are made between *application domain, perspective*, and guidelines on *techniques, tools, and principles of organization* when analyzing system development methods. I find this framework useful in order to make theoretical assessments of the approaches described in literature.

2.1 "The Prototype becomes the system" approaches

This category covers approaches where initial analysis and design activities produce a prototype that is stepwise refined into the final system based on a few rounds of user evaluation of the prototype. Descriptions of such approaches can be found in e.g. (Boar 1984, Boehm 1984, Canning 1981, Lantz 1986, Lie-Nielsen & Colliander 1986). The labelling of this category is inspired by the often repeated statement that "The prototype become the system" from (Lantz 1986, p. 5). Other authors such as (Floyd 1984) denote this category as *incremental* or *evolutionary* prototyping approaches depending on how fixed the overall design ideas are.

The *application domain* of these methods is typically development of Information Systems consisting of large databases implemented on mainframes with terminals and possibly PCs. Some of the literature in this category gives guidelines dealing with the full 'life cycle' of computer systems. The underlying *perspective* of the methods represents the project managers and partly the system designers point of view on the design process. The perspective can be illustrated by citation of statements of the typical goals of the approaches. These goals are e.g. to: "reduce development costs", "please users" and "decrease communication problems" (Lantz 1986).

The prototyping *techniques* in this category of approaches are mainly focusing on how to supplement or substitute parts of a traditional requirements specification (e.g. verbal or data-flow descriptions) of a future application with a prototype of the user-interface aspects before the stepwise implementation begin. According to common technology in the application domain (mainframes and terminals) the prototyping activities are typically focusing on developing form-based screen dialogues and/or hierarchical menus for the system. The data content and the expected functionality of the future system is recommended to be thoroughly analysed in advance to the prototyping activities ("Prototyping will be chaotic unless it is based on a good knowledge of the information. You should understand at least 80% of the content and structure of the data." (Lantz 1986, p. 97)). It is a major goal of these approaches that a user interface prototype, often denoted a horizontal prototype[2], can be reused in the final implementation of the system. The implementation of the system can possibly take place as stepwise supplementing a horizontal prototype with vertical prototypes. It is also a goal to involve users in the *evaluation* of the prototypes, but the prototypes are developed with a fixed overall design in mind and they are most frequently developed by the designers on their own. The users comments are expected to be on details compared to the overall structure of the system.

The *tools* that are recommended in these methods are Application Generators (see e.g. (Horowitz et al. 1985, Martland et al. 1986)), 4th Generation Systems (see e.g. (Martin 1983, Martin 1984, Martland et al. 1986)), and relational DBMS. The tools are used to develop horizontal and perhaps vertical prototypes that can be elaborated to constitute the final system. Description tools (e.g. data-flow diagrams) and traditional programming languages are recommended as a supplement for attacking certain problems, e.g. specification of functionality and improvement of performance.

Regarding the *principles of organization of projects* it is recommended to substitute intermediate phases of a traditional waterfall model with iterative activities aiming at stepwise derivation of *the system* from the prototype. It is typically recommended that the design group consist of only 2-3 persons. Different strategies for choosing user representatives for the evaluation activities are typically discussed , too.

Critique

Although I do not possess literature that describe any rigorous use of these methods I will give a partly theoretically and partly empirically based critique of the idea behind these methods. The recommendations given in the literature show that the prototypes developed are only aimed at becoming a means for the designer to provide a more concrete descriptions of the future system, i.e. the designer is able to better demonstrate how the future system will be. This imply that the prototype become a means to *adjust the users horizon of expectations* about what the developers are able to provide, rather than a means for users and designers to create visions on new design solutions leading to a system tailored to the users needs. It is recommended that the users have the opportunity to test and evaluate early versions of "the system". But it is only mentioned in a side remark that users under some circumstances can participate in the development of user interface parts of prototypes, if high-level tools are being used. A prototype is typically viewed as part of the future system, "The prototype become the system" (Lantz 1986, p. 5), instead of a rough sketch that possibly could be thrown away in favour of an alternative solution. Alternative system designs are discussed and decided upon early in a project without previous experimentation, and large parts of the specification is made before the prototyping activities begin. These decisions are made during discussions of abstract descriptions that users hardly can relate to their work tasks.

This theoretical critique can be supported by empirical studies (Christensen et al. 1987, Grønbæk 1988) that describe examples from practice of IS development where 4th Generation Systems were used. In these examples horizontal prototypes were used for presentation only, and even though the users had access to the prototypes in their workplace they were not encouraged to evaluate them, e.g. they got only few resources (such as spare time) and no test examples to play with. The horizontal prototypes only got the role of a substitute for parts of traditional requirements specifications. Prototypes illustrating alternative solutions were not developed, the only aim of the prototypes was to test the adequacy of the solution provided. Vertical prototypes that enable realistic test were developed only in few cases and simulated functionality was not utilized at all in the examples. Thus experiments with prototypes in work-like situations were only organized in a minority of the projects and then quite late in the development process. One of the reasons given by the designers was that providing functionality for a prototype still requires some effort, and they wanted to have a frozen specification before spending time on programming functionality that allows work-like experiments. This lack of experimentation in work-like situations and lack of active involvement of users in the design process implied the usual consequences during the introduction of computer systems in organizations: The users raise a number of new requirements on the "final" systems when they discover how it impacts their work practice and organization. In many of the project examples this lead to unexpected iterations on the design and implementation of the system. Thus development productivity improvements and user satisfaction benefits were not obtained in general.

2.2 Executable specification approaches

This category covers approaches to iterative design and verification of complex program functionality through development of formal specifications in specification languages that are executable and thus capable of simulating the specified program. These approaches are deeply rooted in the software engineering tradition and descriptions are, e.g. found in (Hekmatpour & Ince 1988, Squires et al. 1982).

The *application domain* of these approaches are usually specification of detailed functionality, such as handling of complex data structures, or flow of interaction in certain types of user interfaces. These approaches aim to obtain a fully formal specification of (parts of) the future system. Recommendations on other aspects of the system development process are seldom given. The *perspective* of these approaches is purely the software engineers' perspective on how to develop reliable software. This can again be illustrated through citations of goals formulated in the papers: "making system design concise and productive", "ensuring that a precise level of documentation is available" (Hekmatpour & Ince 1988).

The main guidelines for prototyping *techniques* here deal with how to obtain a precise and fully formal specification of (parts of) a future system. The specification is made in a formal specification language that is executable, and by which a working prototype can be generated. In these approaches the process is an iteration between a formal specification an test of the automatic generated prototype. It is pointed out that users can evaluate the generated prototypes, but it is not discussed how they can be involved in designing it. The ability to prove program correctness is also considered important in some of these approaches.

The *tools* being used are e.g.: formal specification languages such as e.g. VDM, aimed at specifying detailed functionality; graphical specification tools, such as extended STD and data-flow diagrams, aimed at specifying flow of interaction with a system or flow of data within the system; and finally grammars or automatons aimed at specifying command languages. Moreover, a combination of tools to support executable specification approaches for multiple aspects of applications is described in (Hekmatpour & Ince 1988). Finally a new class of tools, the so-called CASE-tools (see e.g. DATAMATION July-1, 1988 for a number of articles on CASE-tools) supporting data-flow oriented system development methods, is now emerging in IS development organizations. Some of these tools support executable specification approaches to prototyping because they are capable of generating, e.g. Cobol or SQL code from data-flow diagrams and structure-charts (see e.g. (Yourdon 1982) for a description of a data-flow oriented method).

Regarding *principles of organization* the literature on these approaches do not say much. The main point of the methods is to obtain a fully formal specification of a system and maintain that specification through a number of evaluations of the automatically generated prototypes.

Critique

The automatically generated prototypes are usually not suitable as parts of a future system, because of lack of integration of the executable specifications systems in implementation environments. Thus the approaches have for many years mostly been applied in research like environments where it has been considered crucial to make formal specifications and correctness proofs of programs. Empirical material describing user involvement in development processes, where executable specification approaches are used, is hardly available. In (Squires et al. 1982) most of the 40 papers discuss approaches in this category, but none of them describe how the prototypes are used for design and evaluation in cooperation with users. This lack is also apparent in (Hekmatpour & Ince 1988) even though examples e.g. describing development of a library system is given. The focus in the examples is on the specification of the system, and not on how the process of prototype design and evaluation is done with users. Thus the examples neither verify that the presented approach can be used to involve users in the design, nor do they verify that final applications for an organization can be generated, because both concurrency and large database aspects are neglected in the examples. Moreover, from a theoretical point of view these approaches seem to suffer from problems similar to those described in the critique of section 2.1. The idea of establishing work-like experiments with the prototypes do not get much attention, because most of the executable specification systems, are too limited to provide prototypes with both user interface and functionality parts suitable for work-like experiments. Moreover, the users can hardly be involved in the design process, because the specification tools are too hard to explain to non-specialists, and the prototypes are manipulated purely through such specifications. Thus the users only have the opportunity to get the role as evaluators of the generated prototypes.

2.3 Exploratory approaches

This category covers a wide range of examples of exploratory experimentation with prototypes for new human-computer interfaces mainly in research settings and seldomly in IS development projects. Descriptions of such approaches are, e.g. found in (Bødker et al. 1987, Hsia & Yaung 1988, Mogensen 1985, Thomsen 1987). This literature does not represent a set of methods, it rather describes examples of innovative design using (manual or software) mock-ups or "throw-away" prototypes.

The *application domains* of these approaches are typical research like design of new interaction styles based on e.g. graphical user interfaces, new I/O devices, and even command languages. The scope of these approaches is usually limited to the early phases of a development process, and the aim is to obtain a basis for requirements specification and/or further development through experiments.

These approaches are taking the *perspective* of a designer as an experimental researcher rather than a rational analyst. The perspective has been denoted as process-oriented (Floyd 1984), which emphasize that the process of building and experimenting with the first prototypes is often more important than the resulting prototype itself.

The recommended *techniques* are aimed at making quick-and-dirty "sketches" of selected user interface and functional parts of a future system. These sketches are used to animate the visual appearance of the system or to simulate functionality to illustrate specific example tasks prepared for user evaluation early in a design process. These approaches do not require a formal specification or a thorough paper-based analysis in advance, only a rough formulation of the overall goal of the development is needed. The techniques are aimed at clarifying requirements before going into the targeted design and implementation a system.

Various *tools* have been applied to develop mock-ups and throw-away prototypes rapidly early in a development process. Tools to create mock-ups made of paper, wood, colour slides etc., have been used (Example given in (Bødker et al. 1987)). When it comes to computer-based tools, especially Lisp and Smalltalk-based tools, have shown to be quite attractive in this kind of prototyping because of the great flexibility. See e.g. (Henderson 1986, Thomsen 1987), for examples on the use of such tools for development of screen-layouts supplemented with simulated functionality. Also specialized scenario-generators, see e.g. (Hsia & Yaung 1988), which are capable of animating a sequence of screen-layouts with prepared data contents have been developed to facilitate early experimentation. Finally human simulated functionality have been applied to the design of voice recognition systems and command languages (An example is given in (Good 1984)). Human simulated functionality is also denoted as Wizard of Oz techniques[3].

Critique

Regarding this category of approaches empirical descriptions of examples such as those in, e.g. (Bødker et al. 1987, Good 1984, Mogensen 1985, Thomsen 1987) exist. In these examples "throw-away" prototypes or mock-ups have been developed and used early in development processes to explore new ideas for both functionality and user interface issues in a work-like setting. The examples where computer-based tools have been used have shown quite promising with respect to the ability to provide hands-on experience to users early in the design process. Users are, however, not actively involved in the design and modification of prototypes. In contrast the examples (Bødker et al. 1987) where noncomputer-based tools were used have shown promising with respect to let the users participate creatively in the development and modification of mock-ups, because the tools to manipulate these are comprehensible to both users and designers. But with these noncomputer-based tools it is hard to provide functionality enough to make the users' imagine that they are in a work-like situation when they are using a mock-up. A disadvantage of these approaches, independent of the use of computerized or noncomputerized tools, is that the gap between the developed mock-up/prototype and the final system can be rather large and thus create misleading expectations among the users on how fast the designers can develop a final application. However, the main advantages seen in the examples are that both hands-on experience with early prototypes and participation in development of mock-ups improve users' engagement and opportunities to influence the system design process. In the next section I will propose an approach that combine the ideas of using from these various exploratory approaches and provide users with both hands-on experience and and better opportunities for participation in the design.

2.4 Summary of Critique: Challenging current prototyping approaches

The first two categories of prototyping approaches are taking a pure software engineer, manager, or designer perspective on system development. Only the last category of approaches is showing trends that consider a users perspective, and thus usability of the systems, to be at least as important as, e.g. program correctness and development costs. In contrast a users' perspective would rather be to focus on their work processes, division of labour, skills required for new work tasks, how interaction with the new system will be like, etc, and thus require usable and tailored computer systems that support their work in a better way. In line with the ideas described in (Winograd & Flores 1986) and (Bødker 1987a) a tailored system of high quality can be characterised as a system that enable users to improve their work and perform it with a minimum of breakdowns caused by the use of the system, i.e. the system should be as much as possible a transparent tool for the user. Of course program correctness is a prerequisite to get few breakdowns, and low development costs are important, too, from the point of view of users who want a new system. But these issues are usually not the main concerns when it comes to user evaluation of a new system design, then the focus is on how future work with the system will be like.

When it comes to development of high quality systems that meet the users' expectations current approaches to prototyping share a number of drawbacks similar to traditional system description approaches. This holds in particular for the most widespread applied category of prototyping approaches practice as pointed out in section 2.1. The main reason to this is that the approaches still let the users have a quite passive role in the design process. The users are acting as evaluators of technical proposals, made by designers on their own, rather than active contributors to the design of the computer system that will influence their future work-place. The users do not get a chance, e.g. through work-like evaluation experiments, to anticipate some of the potential breakdowns the proposed system can cause, or to feel how the proposed system will change their workplace and possibly the whole organization.

Increasing user demands on high quality computer systems

I will briefly argue that passive involvement of user' is insufficient to meet the challenges of both current and future development of computer systems. The reason is that a number of factors contribute to increasing user demands on getting access to computer systems that are tailored to the particular requirements of an application domain. Firstly, the increasing use of computers in organizations and private homes imply that users in general become more familiar with computers and different styles of interaction. Thus we may expect more qualified contributions from users regarding design of computer systems. Secondly, people get more dependent on computer support for the tasks they are performing in their daily work. Most users in the future will know of a number of bad systems that they have either used themselves or heard of from colleagues. Thus we may expect that users in general get more interested and demanding when (if) they get the opportunity to discuss design of a new computer system. Thirdly, a number of computer applications in corporate businesses, such as budgeting, database maintenance and retrieval, statistics, and reporting, become widely supported with standardized program packages. These program packages even make some users able to help themselves with much of their need for computer support in these domains. Thus requests raised for in-house designers or consultants will more frequently be in new application domains where goals often are quite uncertain from the outset. Finally the technological innovation as represented by, e.g. graphical workstations or PC's in networks provide a new potential for variation of options in the design of computer systems. In particular new interaction styles give better potentials to support the variations and complexity of users work. Thus the technological innovation itself creates another dimension of challenges for system designers. The number of optional alternative interaction styles to choose at certain point in a design process increases dramatically. This increase of options in design also add to the importance of experimenting with different solutions by envisioning future work situations exploring the relevant options.

Trends already showing increasing user demands in system development organizations are described in (Friedman & Cornford 1987) on the background of an international empirical study. These trends raise a lot of challenges for system developers on how to improve their work practices. Some of these challenges are of course on technical issues. But an important challenge is to consider how to involve users actively and creatively throughout the development process in order to improve the usability of future systems, because this kind of involvement is the key to ensure that new computer systems are well-tailored to the users needs and support the development of their skills utilizing computer support. I will in the next section outline ideas on how to meet the challenges of user involvement by extending the boundaries of prototyping into a wider range of techniques than represented by the current mainstream of prototyping approaches described in the previous section.

3 Extending the boundaries of prototyping

From a point of view, where the goal of system development is to create well-tailored systems for users, I will, inspired by (Bødker 1987b), propose an extended view of what prototyping should be like. Design and thus prototyping is a process in which designers and users in cooperation determine and create the conditions that make a computer system improve the quality of users' work. To find out how a future computer system improve users' work, the future users must be able to somehow experience this, i.e. to envision the future use situation. To envision a future use situation is *neither* to read a description of a computer system or of its use, *nor* is it to watch a demonstration of a prototype: Human beings possess skills that are not easily articulated or imagined without actually being in a situation of acting. For example, humans normally use tools in the work process without being aware of them, skills of using the tools are usually not articulated, they are triggered in the work situation when dealing with a certain piece of material.

To make such skills contribute to the design of a computer system we have to try out the skills by actually acting in a situation.

Thus good design means that it is important to set up situations which make it possible for the users to gain hands-on experience with tangible models, such as mock-ups or prototypes, modeling the future computer application early in a development process. While being "in the future" conducting a future work task by means of a tangible model certain breakdowns will occur, by which the fluent conduction of the work task will stop. As an example we may become consciously aware of the tool that we are using because it does not fulfill its purpose. In a prototyping process such a breakdown may lead to a change of the prototype, and eventually to a change of the overall design of the future computer system. Refer to (Winograd & Flores 1986, Bødker 1987b) for a discussion of the breakdown concept.

Moreover, one of the most serious critiques that was addressed to the main categories of prototyping approaches was that they did not consider it important to experiment on alternatives during prototyping activities. Hence, the methods create a *blindness* of design much similar to traditional structured description methods. See (Winograd & Flores 1986) for a discussion of blindness in design. In a prototyping process, experiments with alternative prototypes should be set up to create conditions to discuss visionary "What if .....?" questions on a more concrete basis. Consideration of alternative design proposals is in general valuable, because the confrontation between contrasting and perhaps conflicting ideas stimulate innovation. The need for consideration of alternatives has been argued elsewhere in (Grønbæk 1988).

Thus the goals of developing new prototyping tools and techniques should on the one hand be to make it possible for users and designers to envision a future work situation with a new computer system that is expected to be the result of a particular design idea, and on the other hand to provide support for them to play with alternative design ideas early in the development process. System designers need to extend their view of what prototyping is, i.e. prototyping should be more than just having a few users test and approve on an early, incomplete version of "the system". Prototyping approaches having their roots in the category of exploratory prototyping approaches described in section 2.3, should get more attention in general, and in particular in innovative development projects, where ideas are fuzzy and there is a request on getting a tailored system rather than a standard system. But to get more benefits from the exploratory prototyping approaches it is also necessary to allow the users to play a more active and creative role in the prototyping activities. From the designer perspective this is necessary, because the users' skills are the key to develop a well tailored system that fits their needs. From a user perspective this is necessary, to get a feel for how new technical possibilities can or cannot support their work, and to get the opportunity to *influence* the development of computer support for their work.

3.1 Cooperative prototyping

I will introduce a prototyping approach, denoted *cooperative prototyping*. This approach aim at establishing a design process where users are participating *actively* and *creatively* in prototyping in cooperation with designers. The idea of the approach is to rapidly develop one or more tangible models, mock-ups or prototypes, of the future system that can immediately be related to core tasks of the use domain. To use these, possibly alternative, tangible models the idea is to set up sessions with highly work-like evaluations of the models. Breakdowns, caused by a bad or lacking design, are turned into immediate modifications of the models. Breakdowns occurring due to lack of training are handled by further training. This process continues until the group decide that they have a sufficiently clear idea of what the system should be like, and the designers have an idea of how it can be implemented. This kind of process is conducted by a group of designers in cooperation with a group of users, and the process is characterised by a close coupling between design and evaluation activities. Cooperative prototyping requires that the group of designers and users have access to tools that support rapid development and modification of tangible models. The models should provide some minimum of (simulated) functionality that makes it possible to envision future work tasks. This requirement determines *a need for developing computer based tools* to support the process. Modifications should either be made on-line in combined evaluation/design sessions if appropriate, or immediately after the session enabling a new evaluation "the day after". This is to emphasize that it should not last months or years before users get at chance to experience the next version of the prototype.

Cooperative prototyping is meant to combine the ideas of using computer-based tools for exploratory prototyping with approaches to design that allow users to participate in the modification of wood and paper mock-ups as described in (Bødker et al. 1987). There is only few descriptions in literature of examples on design groups both using computer-based tools for exploratory prototyping and having the users participating actively in the sessions where evaluations and modifications are made in a close interplay. An example is given in (Bødker & Grønbæk 1989). Here a colleague and I describe how we used HyperCard - a flexible and cheap design tool available for Apple Macintosh computers - to develop a prototype that made us able to envision a dental case record system in close cooperation with a group of dental assistants through a series of cooperative prototyping sessions. In this case 2-3 users and a designer worked with a single Macintosh workstation in the prototyping sessions. We utilized the direct manipulation facilities of HyperCard to make a number of on-line modifications, when breakdowns occurred in envisioned work situations. When breakdowns occurred in the work-like situation with the prototype, these were either turned into an immediate modification of the prototype, a proposal for later change, or some further instructions on how to use the prototype. In this case connections to remote databases were simulated, and the inherent functions of HyperCard made it easy to experiment on various alternative search strategies, how to utilize a direct representation of the human set of teeth on the screen, and two alternative ways of representing a treatment of tooth in the case record. One of the main conclusions from these experiments is that it was possible with a reasonable small effort (a weeks work), and reuse of existing program scripts to prepare tangible models that were suitable both as means for establishing a work-like situation, and as a means to be extended and modified interactively within the design process.

Another example is found in (Hauerslev & Jakobsen 1988). Here prototyping experiments on the design of a telephone switchboard for a secretary were performed in close cooperation with a user, utilizing a domain specific programming tool built on top of a programmers interface to a network supported graphics package called NeWS running on SUN workstations. In this example a user was evaluating a prototype running on a graphical workstation together with a designer, and modifications needed due to occurred breakdowns were made on-line by a programmer working on another connected workstation in the same room.

These examples are documented elsewhere, and they are good examples of what could be called cooperative prototyping processes. The examples indicate that it may become realistic to perform cooperative prototyping early in a development project. In section 4, I will try to argue this optimism by giving a number of examples of emerging tools that can be used to support cooperative prototyping. Note also that the approach cause no economical trade-offs, because:

On the one hand the approach will lead to development of better quality computer systems, i.e. systems tailored to the users needs. On the other hand it is quite common in current projects to see new requirements raising during installation activities, because systems don't meet the users' expectations. A number of such requirements can be anticipated with experiments early in the development process, thus saving resources at the end, and in the project as a whole. In (Boehm 1981, p. 39 ff.) it is claimed that "errors" detected in the late phases of a development project can be up to 100 times more expensive to correct, than if the same error had been detected in the requirements determination "phase".

Finally, I will note that it is not a demand that all of the prototypes developed with this approach should be thrown away when implementation begins. Actually it would be an advantage to be able to reuse at least a subset of the screen objects, been designed. But to get a cooperative prototyping process going it is important to be able to create and modify quick-and-dirty functionality without being aware of performance issues etc. Thus functionality most likely always need to be programmed anew during implementation activities when applying a cooperative prototyping approach.

3.2 Tangibility and direct manipulation

To undertake a cooperative prototyping process it is important that users and designers are able to work closely together with a minimum of communication problems caused by the prototyping tools being used. The prototyping tools should ideally be understandable, tangible, and transparent[4] to both users and designers as is the case with paper and scissors to make mock-ups, see e.g. (Bødker et al. 1987). But for cooperative prototyping we want to provide some functionality behind the tangible models, enabling *both* a flexible sketching of ideas and evaluation of the ideas in an envisioned use situations. Thus the tangible models have to be computer based and allow immediate hands-on testing. But the tools should still be flexible in the design situation and ideally understandable to the whole group in line with what is the case with wood and paper. This situation is still hard to obtain when it comes to computer-based tools, e.g. a programming language is not easy to understand for a nonprogrammer user and thus not suitable to support user creativity. However, tools that provide support for *direct manipulation*, i.e. a point-and-select interface for manipulation of objects in a computer system[5], of computer based tangible models are emerging. In cooperative prototyping situations involving non-computer expert users, direct manipulation tools have a number of appealing advantages compared to conventional programming. Manipulation of objects become visible and on the surface easy to follow. Moreover, immediate feedback shows the result of an operation. Some of these tools are suitable for developing tangible models that both provide functionality to be tried out and the ability to be manipulated directly in an iterative process of cooperative prototyping.

A variety of tangible models, denoted mock-ups, simulations, and prototypes, has been applied for exploratory prototyping as described in section 2.3. But in practice designers only develop targeted prototypes that "become the system", as mentioned in section 2.1. This can mainly be explained by the fact that the widely spread 4th Generation Systems and Application Generators support this kind of prototype development. These tools are usually limited to support direct manipulation of form-based dialogues and report lay-outs. In the next section I will illustrate tools to support direct manipulation of a much wider range of tangible models than seen currently in practice of system development.

4 Computer support for cooperative prototyping

Reminding that it is necessary to have computer support for cooperative prototyping, to simulate functionality for establishing work-like evaluation, I will give a brief survey of sources for developing such computer support. In (Bødker et al. 1988) a set of requirements for tools to support cooperative design is stated from a discussion on properties of cooperative design situations. In this paper the importance of having access to Interactive, Integrated, and Incremental tools is highlighted. Moreover, the tools should ideally be provided as domain specific substrates[6] embedded in a general Object Oriented development environment. I agree on those general requirements, but from the discussion above and practical experiences (described in (Bødker & Grønbæk 1989)) with cooperative prototyping I find it worth highlighting the requirement for direct manipulation support. I will briefly discuss a number of different computer-based tools that seen from this point of view could support cooperative prototyping. Noting that an orthogonal classification of such tools can hardly be given, I group them with respect to the techniques they support using the following labels: Animation, Simulation, Wizard of Oz, Design by Example, and Visual Programming.

4.1 Animation

A number of tools (e.g. screen editors in 4th Generation systems[7], OSU[8]) are capable of supporting design of screen objects such as buttons, windows, and form-based dialogues by simply drawing figures, writing text, and placing fields and icons on an empty screen or a window. When it comes to the functionality behind such screen objects a switch to another environment is often required and also a larger programming effort is required to implement or simulate functionality. However, the kind of functionality that consist in making relationships between the various screen objects can with some tools be illustrated through animation facilities.

When dealing with purely form-based dialogues some screen generators such as SQL-FORMS of ORACLE[9] provide a stepwise animation of a sequence of dialogues by using e.g. function keys to trigger jumps between already prepared screen layouts. Data-items can be entered in the fields of the screen layouts, and they survive the animation steps within the same session with the program. Such early designed screen layouts can as a result of a number of iterative modifications later on be used for the implementation of the system. In (Hsia & Yaung 1988) a dedicated screen-editor system is presented, which is capable of both having data items entered and specifying simple branching conditions for the animation of a sequence of dialogues. In the latter case the screens and their sequencing, here denoted a scenario, is not integrated in a system that allow reuse for implementation.

When dealing with graphical user interfaces with buttons, dialogue boxes, pop-up menus, etc., similar kinds of animation facilities can be provided. One option is to use a dialogue editor design the dialogues and a macro recorder to implement the sequence of dialogue appearances by recording the steps involved in bringing the various dialogues up on the screen. Another option is provided by an experimental development environment called OSU[10]. Here various sorts of interface objects for the Macintosh user-interface, can be designed by direct manipulation. A proposed sequence of appearance of the objects for an application can be designed with a so-called Graphical Sequencer by inserting the objects in a graph specifying the sequence of dialogues and the triggers to step through the sequence. From the sequence graph a Pascal shell program can be generated, and the whole sketch can be used as input for the development of a final application in a CASE-tool like environment.

The tools described support iterative development of fully animated horizontal prototypes. They are highly interactive and direct manipulation based, hence they could support some of the design aspects of cooperative prototyping. The steps in developing and manipulating the screen-objects and the sequences consisting the design proposals requires little effort, and keep the need for paying attention to technical matters to a minimum during the design process. But evaluation aspects such as establishing work-like situations with the prototypes is poorly supported with the limited kind of functionality that is available with these animation tools.

4.2 Simulation

This label covers use of design tools that both support direct manipulation design of user interfaces and simulation of functionality. Simulation in this context means ability to provide quick-and-dirty program functionality behind the visual and tangible parts of a user interface. A program simulation can e.g. be used to illustrate how machines, such as photo copiers or waste incinerators, are functioning while the user interface for a control system for the machine is being designed. Program simulations can also be used to develop quick-and-dirty functionality behind IS prototypes, e.g. access to remote databases can be simulated with a small local database, and data entry can be accepted and stored without detailed validation.

To give an example of simulation of a machine it is worth mentioning Trillium, described in (Henderson 1986), which is a substrate built in Interlisp-D to support design of photo copier and printer user interfaces. With this substrate it is possible to build user interfaces consisting of buttons, controls, displays, and meters through direct manipulation. Trillium also supports creation and reuse of objects that are aggregations of the primitives. Behavioral dependencies between the objects on a corresponding physical machine are automatically simulated with an underlying program in order to be able to let users/designers envision that they work with a real photo copier when using the simulation. Trillium is a domain specific tool and supports only the design of interfaces to "simple" machines that have interfaces similar to photo copiers. As such Trillium has had success as a tool for prototyping such user interfaces in the various groups of designers working with photo copiers and printers within Xerox Corporation, because of its ability to support experiments with functionality of an interface without building the much more expensive physical machine.

A number of 4th Generation Systems, e.g. MANTIS[11] for IS development provides some support for simulation of functionality behind horizontal prototypes. In these cases the screen editor allows the designers to connect the fields from the screen layouts to a data structure with search and sorting facilities that makes it possible to store and retrieve a small number of test data items without programming functionality based on the larger and less flexible database system. These facilities makes it possible to support experimentation with some work tasks using modest test data sets early in a design process. In other 4th Generation Systems early functionality can be provided utilizing the integration between a screen editor and a relational database system. In some systems such as DATAFLEX[12] relational tables can be generated automatically from screen layout specifications or vice versa. In these cases it often requires more effort to provide functionality and the boundaries between a simulation and a vertical prototype become blurred.

HyperCard for the Macintosh is an example of a combination of a database and hypermedia tool that allow design of graphical user interfaces and hypermedia applications exploiting fields, graphics, buttons, menus, and simple dialogue boxes. The basic metaphor to understand Hyper-

Card is a cards-in-boxes metaphor. A lot of functions that manipulate cards and data contents entered on cards are available as standard. HyperCard is not suitable for building large scale IS systems, but it provides some nice facilities to simulate parts of such systems within certain application domains, such as medical case records where the existing manual records are built up of a number of sheets or cards that are contained in folders and boxes. In (Bødker & Grønbæk 1989) an instructive example of the usage of HyperCard for cooperative prototyping of a case record system for dental clinics is described. In this case it was shown to be possible to reach a design process with a close coupling between a fluent work-like evaluation of a prototype and on-line modification of the prototype. See also section 3 for a brief description of the example.

4.3 Wizard of Oz

In (Wilson & Rosenberg 1988) a so-called Wizard of Oz[13] technique is described, it is based on the idea of having "a man sitting behind a curtain" to simulate functionality of imagined programs. These techniques have been applied in domains such as design of speech recognition and command language user interfaces. In these domains the techniques only require a modest set of tools. The whole idea is that a new command language is designed through experiments where a user either talk to a microphone or type commands on a terminal, and in both cases the user input is interpreted by a designer or an operator who turns the commands into a reasonable and immediate feedback from the computer. From this process users and designers can get and idea of which commands and functions it would be natural to have for a certain use domain. An example of such experiments on the development of a better command language interface to an electronic mail system is described in (Good et al. 1984). Here it was concluded, among other things, that adding numerous synonym commands to the mail system improve both expert and novice satisfaction with the system. Using this technique with a designer/programmer acting as imagined computer functionality could be a way to establish a cooperative prototyping process on certain aspects of a new computer system. The human simulated functionality is quite flexible to modify, it is understandable to the participants, and experiments in work-like situations are clearly possible (refer to the case on electronic mail mentioned above).

The idea behind this technique is quite exciting for the discussion of cooperative prototyping. But unfortunately it is hard to extend the technique to simulate functionality behind full-screen and graphical user interfaces with a designer/operator sitting at another workstation. It will be hard for the designer/operator to interpret what a user expect to happen, when e.g. an icon is painted on the screen and then clicked. The designer need some way to "read the users mind" during the experiments. One possibility for doing that is to have the users' verbally expressing their expectations on the functionality for the designer in each step of the interaction. The overhead of verbal expression, however, makes the imagined use situation less realistic compared to the use of the technique for command language design. Another possibility to get a work-like situation going with human simulated functions is to agree on the functionality behind certain buttons and menu items, and then have the operator receiving a signal on his/her terminal telling when this particular operation is requested from the user. The imagined system response will, however, also be harder to simulate for the designer "sitting behind the curtain", because windows, icons, etc. often need to be positioned according to the users' pointing with a mouse. The designer need to somehow have a "hand" working on the users screen. More work on these ideas need to be done in order to explore if, and how the techniques can be applied to cooperative prototyping of systems with interactive full-screen and graphical user interfaces.

4.4 Design by Example

This label covers tools to support rapid development of tangible models in a manner that is often denoted "Design by Example" or "Programming by Example". Such tools support designers and perhaps end-users in designing user interfaces or final applications in certain domains through stepwise demonstration of how the user interface or application should behave. These tools are closely related to the visual programming tools that I discuss briefly in the next subsection.

An example of a domain specific "Design by Example" tool can be found in the IS domain, where a number of tools, to create reports from existing databases through specification of an example report in a template or a form, have been marketed. An example of such a tool is Query By Example (QBE), see e.g. (Gray 1984, pp. 145-154). QBE is a query facility that support information retrieval e.g. report generation from databases by means of template filling. When the users want to request a report from a (relational) database a template showing the attributes of the requested table is brought up on the screen. Then the user have to describe an example of the criteria that each attribute should fulfill, afterwards the report output is automatically generated with a standardized layout usually determined by the system. To make retrieval from single tables the user need some familiarity with the inherent structure of the database and mathematical symbols such as "=", "<", ">", "not", "and", "or" to express conditions. To make combinations from a number of tables a more advanced command syntax describing attribute selection from different tables is required. These tools can support cooperative prototyping for this particular use domain, because traditional programming can be kept to a minimum in prototyping sessions as far as the standardized layout styles are satisfying. Assuming that an example database can be provided these tools also allow work-like evaluation of prototypes.

An example of a more general "design by example" tool is a UIMS[14] for graphical user interfaces called Peridot, which is implemented in Interlisp-D, and described in, e.g. (Myers 1987). Peridot allows the designer to draw various types of screen objects in a display window and make them immediately usable for interaction as is the case with a number of the tools described above. The special feature of this tool then is that it allows the designer to "program" the dynamics of the user interface by demonstration of the users mouse movements and clicking on menus, buttons, scroll bars, and windows etc. A virtual mouse with three buttons appearing as an icon on the screen is the representation of the real mouse. The moving of this icon on top of e.g. an already drawn menu simulate the use of the menu. This way It is possible to specify, e.g. whether an item is chosen when pushing one of the mouse buttons or when a mouse button is released on top of an item, also toggling and inversion of a menu item can be demonstrated. The interfaces created with Peridot can be tried out immediately as stand-alone or together with the application program that the user interface is designed for. Moreover, rules of conventions from previous demonstrations are recorded automatically in a rule base and then reapplied when new objects similar to existing objects are created.

Myers who has developed Peridot claims that "Peridot can create almost all of the Apple Macintosh interface, as well as many new interfaces such as those using multiple input devices concurrently" and "Nonprogrammers can use it to create user interfaces" (Myers 1987, p. 59). If that holds it means that Peridot is an extremely powerful tool for designing graphical user interfaces without traditional programming activities. Thus the ideas of Peridot seem to be a potential source for developing tools for cooperative prototyping. A problem to be solved is, however, that the user interface design is still seen from the technical point of view by having menus, windows, etc. as the primitives. In a situation where users and designers work together on designing an application for a certain use domain it is also necessary to aggregate and specialize the primitives into objects that correspond to objects well-known from the use domain in order to support the users understanding of what is going on. To turn a powerful tool such as Peridot into a tool for cooperative prototyping within a certain use domain, a "palette" of domain specific objects needs to be accessible as a library.

4.5 Visual programming

This final group of tools that can be considered as a potential source to provide computer support for cooperative prototyping is denoted visual programming systems. The term visual programming is expressing that such systems are attempting to substitute traditional programming activities with point-and-select programming through diagram manipulation, form-filling, menu-selection, and clicking of icons. With regard to the earlier systems mentioned in the discussion, this is not a totally new group, actually a number of the systems mentioned above, e.g. OSU, QBE, and Peridot, have occasionally been denoted as visual programming systems, because of their substitution of traditional programming with direct manipulation programming. Recently work has been done on developing fully visual and general programming systems to substitute traditional programming languages. An example of such a system is an object-oriented programming environment called Prograph[15]. This environment provides: direct manipulation design of screen objects, fully iconic/graph based program editing, graphical browsing to edit specialization hierarchies, interpretation based execution, and debugging based on animation of the diagram/icon program, visual showing of the runtime stack, and finally support for persistent objects. These features makes Prograph intuitively appealing to a programmer/designer, but beyond user interface design, programming in the environment is still involving a lot of low level programming details, although it is done through drawing of lines and placing of icons. This fact makes it unlikely to be a good idea to have users involved in programming of system functionality with Prograph. But the highly interactive graphical browsers and editors together with interpretation based execution could potentially support cooperative prototyping with on-line modifications based on reuse/specialization of existing domain specific classes/objects embedded in the visual programming general system. Moreover, persistent objects support a convenient handling of example data that is needed to establish a work-like evaluation of prototypes.

4.6 Overloading direct manipulation ?

In the previous sections a number of direct manipulation design tools and their potentials to support cooperative prototyping, have been discussed. I have found this kind of computer support quite attractive and promising for this purpose. But it should not be ignored that direct manipulation tools also share some drawbacks. Since Shneiderman (1983) introduced the concept of direct manipulation and named it to be one of the most promising approaches to development of human-computer interfaces some authors have been critical against the ideas of transforming all sorts of human-computer interaction into direct manipulation. In (Norman & Draper 1986, Chap. 5) it is argued that direct manipulation tools show to be tedious and difficult to use in a number of cases such as: performing repetitive operations, representing variables, and providing flexibility and efficiency to expert users. It is also claimed that direct manipulation tools narrows the borders of innovation, and their concluding claim is that general and powerful programming languages never will or should be substituted with direct manipulation tools. I agree that these critical remarks holds for some direct manipulation facilities, e.g. it becomes tedious for an expert programmer to pick up icons for each statement of a programming language, because statements can be typed in faster from the keyboard once you are familiar with the syntax. Hence I do not either believe it is a good idea to turn all human-computer interaction into direct manipulation facilities, but users and designers who want to perform cooperative prototyping need direct manipulation support for more of the programming activities than than is the case for programmers/designers who do programming on their own.

To summarize I believe that development of direct manipulation tools embedded in a general and powerful object-oriented programming environment is a good direction to move when developing computer support for cooperative prototyping. Hence the direct manipulation tools can be used to stimulate *user* and designer creativity in the design and evaluation sessions, and the more powerful underlying programming environment can support other aspects of the designer creativity when designing and extending the set of domain specific objects that can be used in a direct manipulation fashion in later activities of the evaluation and design process.

5 Concluding remarks

In this paper I have argued that the prototyping approaches currently applied in practice give a too limited support with regard to allow user involvement and creativity which is crucial to develop usable computer systems tailored to the user needs. I have argued for applying an approach to prototyping, denoted cooperative prototyping, to stimulate user and designer creativity in particular early in a development process. Moreover, I have given a number of examples of direct manipulation tools and techniques potentially supporting a cooperative kind of prototyping in various use domains. But what is more important about the examples: They should point at sources for further development of computer support for cooperative prototyping.

The fact, that most of my examples utilize tangible models that often cannot be reused in final applications for the respective use domains, does not mean that users should only be involved in the early stages of system development. It rather means that the proposed techniques should be supplemented with a stepwise prototyping approach similar to those described in section 2.1, when the goals of the development have been clarified through cooperative prototyping. It is, however, important to maintain a continuous, but perhaps less frequent, user involvement throughout the full project to be able to pick up on organizational changes etc. occurring while the project is running. To summarize the aim of the cooperative prototyping approach is to get a more concrete medium for the early design discussion in order to allow *both* users *and* designers to be creative at this stage. In the research program "Computer Support for Cooperative Design and Communication", described in (Bøgh Andersen 1987, Bødker et al. 1987) we are working both empirically in different use domains and experimentally in laboratory with the development of tools and techniques that support the kind of cooperative prototyping argued in this paper.

Notes

* Full affiliation: University of Aarhus, Computer Science Dept., Ny Munkegade 116, DK 8000 Århus C, Denmark. Phone: +45 86 12 71 88 ext. 5057. E-mail: kgroenbaek@daimi.dk.

1. The *critical* school according to (Bansler 1989).
2. The notion of horizontal and vertical prototypes are discussed in (Floyd 1984).

3. The name "Wizard of Oz" is due to a novel telling a story about a wizard having a man sitting behind a curtain to help with his acting, see (Wilson & Rosenberg 1988).
4. Refer to (Bødker 1987a) for a discussion of transparency of tools.
5. See (Shneiderman 1983) for an introduction of the direct manipulation concept.
6. Substrate means an "Underlying layer", and in this context it denotes a collection of system objects specialized for a certain application domain.
7. Refer to e.g. (Martin 1983).
8. Refer to (Lewis et al. 1989, Yang et al. 1989).
9. ORACLE and SQL-FORMS are trade marks of ORACLE Corporation.
10. OSU is described in (Lewis et al. 1989, Yang et al. 1989).
11. MANTIS is a trade mark for Cincom Systems, Inc.
12. DATAFLEX is a trade mark of Data Access Corporation, Florida, USA.
13. See note 3.
14. User Interface Management System.
15. Prograph is a trade mark of the Sun Gunakara Systems ltd., Halifax, Canada.

Acknowledgements

I would like to thank Susanne Bødker, Morten Kyng, and the members of working group 7 at the 12th IRIS conference held in Skagen, Denmark, August 1989 for their comments on draft versions of this paper. Finally this work has been supported by *The Danish Natural Science Research Council*, FTU grant no. 5.17.5.1.30.

References

Peter Bøgh Andersen(editor). Research Programme on Computer Support in Cooperative Design and Communication. IR 70, Computer Science Department, Aarhus University, Århus, 1987.

Andrew Friedman and Dominic Cornford. Strategies for meeting user demands. In Gro Bjerknes, Pelle Ehn, and Morten Kyng, editors, *Computers and Democracy*. AVEBURY, Aldershot, 1987.

Jørgen Bansler. Systems Development Research in Scandinavia: Three Theoretical Schools. *Scandinavian Journal of Information Systems*, 1(1):3–20, August 1989.

Bernhard H. Boar. *Application Prototyping - A requirements definition strategy for the 80s*. John Wiley and Sons, Inc., New York, 1984.

Susanne Bødker. Prototyping revisited - design with users in a cooperative setting. DAIMI PB 233, Computer Science Dept, Aarhus University, Ny Munkegade 116, DK 8000 Aarhus C - Denmark, September 1987. In proceedings of the 10th IRIS seminar.

Susanne Bødker. Through the interface — a human activity approach to user interface design. PB 224, Computer Science Department, Aarhus University, Århus, April 1987.

Susanne Bødker, Pelle Ehn, John Kammersgaard, Morten Kyng, and Yngve Sundblad. A UTOPIAN experience: On design of powerful computer-based tools for skilled graphic workers. In Gro Bjerknes, Pelle Ehn, and Morten Kyng, editors, *Computers and Democracy - A Scandinavian Challenge*, pages 251–278. Avebury, Aldershot, England, 1987.

Susanne Bødker, Pelle Ehn, Jørgen L. Knudsen, Morten Kyng, and Kim H. Madsen. Computer Support for Cooperative Design. PB 262, Aarhus University, Aarhus, August 1988. Also published in the proceedings of CSCW 88.

Susanne Bødker and Kaj Grønbæk. Cooperative Prototyping Experiments - Users and Designers Envision a Dental Case Record System . In John Bowers and Steve Benford, editors, *Proceedings of the first EC-CSCW '89*, UK, September 1989. Computer Sciences Company.

Barry W. Boehm. *Software Engineering Economics*. Prentice-Hall, Englewood Cliffs, New Jersey, 1981.

Barry W. Boehm, Terence E. Gray, and Thomas Seewaldt. Prototyping versus specifying: A multiproject experiment. *IEEE Transactions on Software Engineering*, 10(3):290–303, May 1984.

Richard G. Canning. Developing systems by prototyping. *Edp analyzer*, 19(9), September 1981.

Søren Christensen, Kaj Grønbæk, and Tove Rolskov. Arbejdsformer under anvendelse af 4. generationsværktøjer. IR 69, Computer Science Department, Aarhus University, Århus, May 1987.

Christiane Floyd. A systematic look at prototyping. In Budde, Kuhlenkamp, Lars Mathiassen, and Heinz Züllighoven, editors, *Approaches to Prototyping*, pages 1–18. Springer-Verlag, Berlin-Heidelberg, 1984.

Michael D. Good, John A. Whiteside, Dennis R. Wixon, and Sandra J. Jones. Building a userdrived interface. *Communications of the ACM*, 27(10):1032–1043, October 1984.

Peter Gray. *Logic, algebra and databases*. Ellis Horwood ltd., Chichester, 1984.

Kaj Grønbæk. Rapid prototyping with fourth generation systems - an empirical study. DAIMI-PB 270, Computer Science, Aarhus University, Århus, November 1988. Paper presented in a forum on Rapid Prototyping at HICSS-22, Forthcoming as an article in the journal Office: Technology and People.

J. Hauerslev and N. Jacobsen. Sokrates: Sprogspil og kvalifikationer. Masterthesis made at the Computer Science Department, University of Aarhus, In Danish, November 1988.

Sharam Hekmatpour and Darrel Ince. *Software Prototyping, Formal Methods and VDM*. Addison-Wesley, Workingham, England, 1988.

D. Austin Henderson. The Trillium User Interface Design Environment. In *CHI '86 Proceedings*. ACM, April 1986.

Ellis Horowitz, Alfons Kemper, and Balaji Narasimhan. A survey of application generators. *IEEE Software*, pages 40–54, January 1985.

Pei Hsia and Alan T. Yaung. Screen-based scenario generator: A tool for scenario-based prototyping. In Bruce D. Schriver, editor, *Proceedings of the Twenty-First Annual Hawaii International Conference on System Sciences, Vol. II*, Massachusetts Av., Washington D.C., 1988. Computer Society Press of the IEEE.

John Kammersgård. A discussion of prototyping within a conceptual framework. In Budde et al., editor, *Approaches to prototyping*. Springer-Verlag, Berlin, 1984.

Kenneth E. Lantz. *The Prototyping Methodology*. Prentice Hall, Englewood Cliffs, 1986.

T. G. Lewis, Fred Handloser, Sharade Bose, and Sherry Yang. Prototypes From Standard User Interface Management Systems. In Bruce h. Shriver, editor, *Proceedings of the 22nd annual*

Hawaii International Conference on System Sciences, Vol II, Washington D.C., January 1989. IEEE.

Sophus Lie-Nielsen and Lars Colliander. Principer för en ny generation systemutvecklingsverktyg. RDFs Rapportserie 16, Riksdataförbundet, Lund, 1986.

James Martin. *Fourth generation languages Vol. I.* SAVANT, 1983.

James Martin and Joe Leben. *Fourth generation languages Vol. II: Representative 4GLS.* Prentice-Hall, Englewood Cliffs, N.J., 1986.

David Martland, Simon Holloway, and L. Bhabuta. *Fourth generation Languages and Application Generators.* The Technical Press - Unicom Applied Information Technology Report Series, 1986.

Lars Mathiassen. Systemudvikling og systemudviklingsmetode. Publication PB-136, Computer Science Department, Aarhus University, Århus, 1981. Also DUE-report nr. 5.

Klaus Viby Mogensen. Afprøvning af systemudvikling med prototyper. DIKU-report 85 10, Institute of Computer Science, University of Copenhagen, Copenhagen, Denmark, October 1985. Masterthesis.

Brad A. Myers. Creating Interaction Techniques by Demonstration. *IEEE Computer Graphics and Applications*, pages 51–60, September 1987.

Donald A. Norman and Stephen W. Draper. *USER CENTERED SYSTEM DESIGN - New Perspectives on Human-Computer Interaction.* Lawrence Erlbaum Associates, Publishers, Hillsdale, New Jersey, 1986.

Ben Shneiderman. Direct Manipulation: A Step Beyond Programming Languages. *Computer*, pages 57–69, August 1983.

Stephen L. Squires and Martha Branstad and Marvin Zelkowitz, Editors. Special issue on rapid prototyping. Software Engineering Notes, Vol 7. 5, ACM SIGSOFT, Baltimore, December 1982. Working papers from ACM SIGSOFT Rapid Prototyping Workshop, April 1982.

Steen Thomsen. Experiences with system specification by prototyping. Presented at the EFISS-87 conference, Roskilde, Denmark, November 1987.

James Wilson and Daniel Rosenberg. Rapid prototyping for user interface design. In Martin Helander, editor, *Handbook of Human-Computer Interaction.* North-Holland, Amsterdam, 1988.

Terry Winograd and Fernando Flores. *Understanding Computers and Cognition.* Ablex Publishing Corp., Norwood, New Jersey, 1986.

Sherry Yang, T. G. Lewis, and C. C. Hsieh. Integrating computer-aided software engineering and user interface management systems. In Bruce h. Shriver, editor, *Proceedings of the 22nd annual Hawaii International Conference on System Sciences, Vol II*, Washington D.C., January 1989. IEEE.

Edward Yourdon. *Managing the System Life Cycle.* Yourdon Press, New York, 1982.

FAILURE OR SUCCESS OF THE IMPLEMENTATION PROCESS OF AN INFORMATION SYSTEM, AN EXERCISE IN REFLECTION-IN-ACTION

Ari Heiskanen
Offices of the Rector
EDP Planning Office
University Of Helsinki
Hallituskatu 8
SF-00100 Helsinki

"In the world of uncertainty, success is only loosely correlated with effort, and chance can never be ruled out as the main cause of either success or failure." [PRWI79, p. 189]

ABSTRACT

The failure or success of an information system is currently a research topic of many scholars, but the essence of these concepts seems quite elusive. The analysis of failure or success has mainly used surrogate measures. In this paper, I describe a software development and system implementation process and show how the success or failure could be at least partially be operationalized in a specific context. In a theoretical analysis I explain the reasons why the general operationalization is so difficult by presenting an action oriented framework for the success of the development process of an information system. The context of this paper is from my work as an EDP practitioner.

The main dilemma in the analysis of success and failure is that in the beginning of development work the actors only vaguely know the goals and the operationalization of them, and afterwards the evaluation of the outcome may include a considerable amount of rationalization. The comparison to some outside standard is also problematic.

1. Introduction

The failure or success of the implementation and use of an information system (IS) is an important question, but the whole phenomenon is poorly understood [KWZM87]. It is quite a common opinion that most information systems are failures in one way or another [LYHI87]. However, recent Finnish surveys provide quite a positive picture, contradicting at least partially the notion that failure is inevitable: the majority of the users of many contemporary systems are quite satisfied [HHMN88, RAHU88, SASÄ89].

The aim of this paper is to give detailed insight into the problems of defining success and failure in a single case, and thus make more understandable the fact that surrogate measures of success and failure are often used [CARL74, SASÄ89]. Empirical data is drawn from a project during the course of which a department level system for processing information about the students of the University of Helsinki is implemented. The findings are put into a theoretical context by proposing a descriptive model for the assessment of the IS implementation process.

The paper is an attempt to apply "reflection-in-action" [SCHÖ83]. Its theoretical framework can be traced to implementation studies [PRWI79, LUCA81, IIVA85, IIVA86], work psychology in the context of IS development and use [SIMI88, NIEM89], theory of goal-oriented practical action in order to change the "status quo" [LDFS44, ARPS87], IS failure studies [LYHI87, LYYT88], and IS assessment [BJDA88].

The specific issues are:

(1) What do different stakeholders of the system mean by failure and success?

(2) What is the framework for the evaluation of success and failure?

2. The Implementation Project

The implementation of the department level subsystem of the Student Information System of the University of Helsinki (SISUH) will proceed in phases where a set of departments will start to use the system. This paper discusses the experiences of the first set consisting of eight departments and one faculty office for its departments. This project (called DIRE, Distributed Registration) started in March 1988 and ended in March 1989 [DIRE89].

2.1 *Background*

The University of Helsinki has eight faculties, 124 teaching departments, about 27,000 students, and 4,200 employees. The Senate, a collegial body of deans and vice-deans of the faculties, and the Rector, are in charge of the administration.

The organ preparing and executing matters of University's administration is the Offices of the Rector, headed by the Rector and the Director of Administration. The Actuary's Office, the unit responsible of the University's centralized student register (ACTRE, Actuary's Register on a VAX-11/780), and the EDP Office, are located in the Offices of the Rector. The department is the basic administrative unit for teaching and research in one or several subjects of study or fields of research. One of the (full) professors of the department is the head, and there is no official department council. Administrative EDP is led by the administrative EDP Leading Group, consisting of the Director of Administration, section chiefs of the Rector's Office, the Head of the Computing Centre, Head of the Planning Office, and the EDP Chief of the Rector's Office.

2.2 *Stakeholders*

Several organizations and groups of people need information about the students. The importance of SISUH is growing because of the use of exact information about the academic progress of students when evaluating the activities of all universities in Finland, as required by the Ministry of Education. The principal bodies interested in the student information system are (1) teachers, (2) clerks of the departments, (3) departments represented by the Head, (4) administrators (including EDP professionals) and (5) students of the University of Helsinki, (6) The Ministry of Education, (7) The State Centre for Educational Grants and Loans, and (8) The Statistical Central Office. During the DIRE project only the first four groups are relevant, because the other ones are not directly affected by the project.

2.3 *The Organization of the DIRE Project*

In March 1988, the Senate set the project by establishing the Project Leading Group, presided over by the Director of Administration and consisting of the heads of the participating departments, and the Dean of the participating faculty, the Head of the Actuary's Office, one section chief of the Offices of the Rector, and myself as the project leader. The resources of the project were approved by the EDP Leading Group. Each department named a person (the department planner) to be responsible for planning the implementation process. In some departments a working group for the project was formed.

The total resources spent in the project were about two man-years for EDP planning and programming. The major part of the programming was done by an independent software firm according to the specifications made within University. The planning work of administrative procedures in the departments has typically required about one person-month per department. The Project Leading Group had five meetings, the planners of the departments had six meetings together with the EDP planners, and there were two general meetings for all persons (totalling up to about forty) within the project. The EDP planners typically visited each department several times, installing software and data for the department, discussing technical features of the system, progress of the project, problems, and the tasks to be performed by the departments. The project included a short course in administrative EDP in spring 1988, and the users were trained to operate the system during autumn 1988. In April 1989 an evaluation seminar was held; all participants of the project were invited (except department heads), but only fourteen persons attended. They were participants from "successful" departments (10), the Head of the Actuary's Office, and EDP professionals (3).

The DIRE project was a one-year effort to improve SISUH, and work is continuing according to the experiences. The specific goals of the project as approved by the Senate were: (1) distributed processing should start in the first departments in autumn 1988, (2) the project should produce a plan to improve the data coverage of ACTRE, and (3) the project should produce a plan for how other departments can efficiently adopt distributed processing.

2.4 *Technical Solution*

The centralized student information system (STURE, Student Registration) used by the Actuary's Office to maintain the ACTRE was technically updated and transferred from the old Burroughs B7800 mainframe of the University to the VAX-11/780 during the years 1985-1987. The new system employs a relational database using MIMER software. During 1987 and the beginning of 1988 a prototype system running on a microcomputer and using dBase III for departments (SIND, Student Information System in Departments) was developed within the framework of an experimental project started in 1986 with six departments, all of them continuing in the DIRE project. The prototype was completed as usable SIND software in autumn 1988, and additional functions were included in the software during the project.

Departments with DIRE could choose alternative ways of processing student data, especially academic progress. The main options (one or several together) were

1 The department continues its traditional manual system of paper files. The transfer of data between the department and the Actuary's Office employs paper forms and letters as the data carrier.

2 The department may adopt the STURE system with centralized data base.

3 The department may adopt the SIND system containing data only on their "own" students.

4 The department uses an EDP system developed by its own resources, optionally in connection with STURE or SIND.

3. Research Methods

The work reported in this paper is a part of a larger longitudinal research effort. The current time frame - about one year - is not long enough to reveal all the effects of the new system, but it is reasonable from the standpoint of the research problems discussed in this paper. The assumptions and the shaping of the research situation closely resemble those of Kling's web model [KLIN87]. The research has features of action research [SUEV78], or action science [ARPS87].

Research is based on the project documentation, questionnaires directed to the participants and answered in October-November 1988, diaries of myself and an EDP planner about the planning situations with the departments, minutes of the project meetings, and questionnaires gathered during the assessment seminar. The goals of the participating persons were recorded in the first questionnaire. The general requirements for the new system, including the official goals of the departments (May-June 1988) and the descriptions of the process (January 1989), were both written by the personnel of the departments. Active and participative observation is also a prominent research method here. The first "formal" assessment of the process was obtained during the assessment seminar in April 1989, but it seems that we must wait until the end of this year to see the main effects of the change process.

In this kind of research there is an obvious ethical problem: how to write the report in a way allowing the essential facts to be reported but the intimacy of individuals to be preserved. The whole DIRE process can, however, be seen as an organizational learning exercise [ARSC78], and thus camouflage should be avoided as much as possible since errors are a source for improvement. This also concerns practice, the openness of the organization, and the way it treats errors. In the same spirit, the questionnaires were not anonymous, and this might have affected the answers. The danger of subjectivity is also great. I am trying to avoid it by relying mainly on written sources, whether public documentation or responses to questionnaires.

4. Defining Failure and Success

4.1 *General Framework*

The relevant every-day meaning of success is: (1) success is the achievement of something you have been trying to do (not failure), or (2) success is the achievement of a high position in a particular field (prosperity). In short, failure is (1) a lack of success in doing or achieving something, or in a particular area of activity (not success), or (2) an unsuccessful person, action or thing (flop) [COCO87].

There is a fundamental difference between the two definitions of success and failure. The first ones take the goal of an actor as the referent point, while the second ones use an "outside standard" as the referent. This difference has a major effect on the assessment methods.

The failure or success of the development and use of an IS has many dimensions, e.g. technical, economic, organizational and social [BJDA88, preface]. In the assessment of an information system (ISA) we can use concepts like acceptance, the rate of the utilization of the IS, performance of individuals and organizations including financial effects, and user satisfaction (cf. [KWZM87]). As Davis and Srinivasan [DASR88] have pointed out, success and failure mean different things to different people. In the area of technology assessment it is observed that failure and success are often evaluated according to quite different scales, e.g. the failure of a nuclear plant may mean sickness and death, while its successful operation means produced electricity, just to name an extreme example [SIMM77]. Dickson et al. [DIWW88] and Davis et Hamann [DAHA88] suggest the use of the following questions proposed by Cameron [CAME80] when assessing organizational effectiveness in connection with ISA: (1) What domain of activity is being focused on? (2) Whose perspective, or which constituency's point of view, is being considered? (3) What level of analysis is being used? (4) What time frame is being employed? (5) What type of data -- or more specifically what variables and what measurements -- are to be used? (6) What referent is being used?

Hirschheim and Smithson [HISM88] define three zones found in the IS evaluation literature: efficiency, effectiveness, and understanding. The first two are more rational and objective while the last one is more subjective and political. They argue that the evaluation should have a more interpretivist perspective where the focus is more in understanding and qualitative research about the social meaning of the IS than in quantitative analysis. Hawgood and Land [HALA88] consider effectiveness to be a subjective concept, because it is seen differently by different people. Even objectivity in the evaluation of the efficiency of an IS is somewhat misleading, because there is always a subject deciding the measures and other criteria to be used (more broadly asking the questions to be answered). Also, the judgement of acceptable levels of some variables describing the IS or its impact is at least partly subjective, although some measures (like response time or money spent on hardware) are more objective than others (like the acceptability of the VDU display lay-out).

Lewin et al. [LDFS44] point out that "success" of human behaviour means in everyday language two different things, firstly the psychological feeling caused by the outcome of an action as experienced by the actor according to its aspiration level, and secondly the objective difference between aspiration level and performance. Lewin et al. [LDFS44] use success to mean the feeling. Accordingly, I argue that in the connection of ISA success and failure must mainly be seen as a psychological feeling of people involved towards the IS. The feeling experienced may be based on hard facts measured from the process against a pre-set goal, or at the other extreme it may be just an illusion. In the following I analyze the situation through a model adopted from the classics of social science [LDFS44] and psychology of work [HACK81]. The framework of the model is from action science [ARPS87].

When acting towards a goal, individual actors set a level of aspiration. Argyris et al. [ARPS87] describe the connection between setting goals and experiencing failure or success as follows:

"... [the] individual experiences [a] sense of success when (1) he is able to define his own goals, (2) the goals are related to his central needs, abilities and values, (3) he defines the paths to these goals, and (4) the achievement of the goals represent a realistic level of aspiration to him."

The goal in action has a structure. It may be an ideal goal, or an action goal. Knowing that the ideal goal ("effective, easy-to-use IS with modest cost") is too difficult to obtain, the actor sets a more realistic action goal, which is thought to be within the available capabilities. The action goal is taken as the criterion for the level of aspiration. As Lewin et al. [LDFS44] point out, the setting of an action goal does not mean that the actor has given up its ideal goal. They write:

"In order to understand this behavior, we must consider the whole goal structure of the individual. This may include quite a number of more or less realistic goal levels. Goal level within one goal structure may include a high dream goal, a somewhat realistic wish goal, the level which the person expects to reach when he tries to judge the situation objectively, and a low level he might hit if luck were against him. Somewhere on that scale will be what can be described as the action goal, e.g. what the person 'tries for' at that time; somewhere his ideal goal will be located."

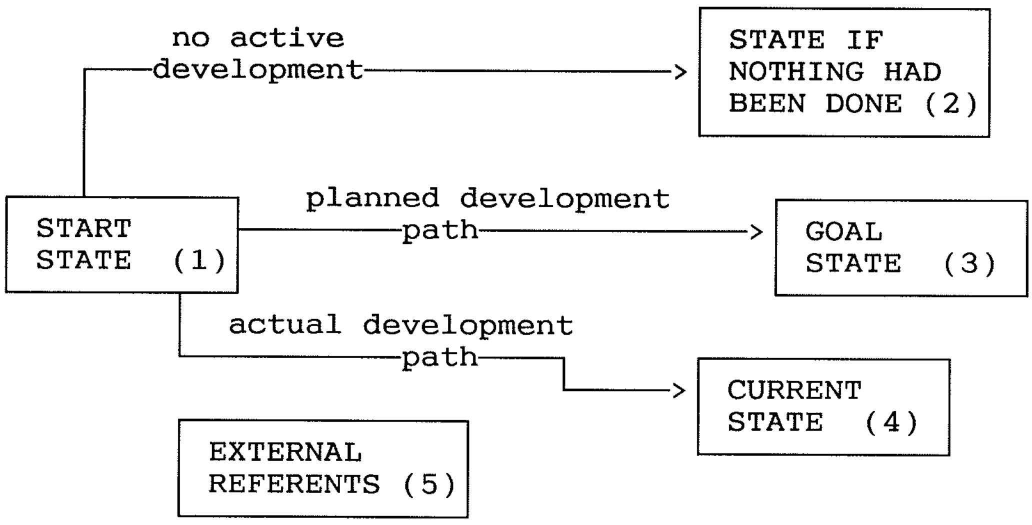
Hacker [HACK81] argues that the performance level and aspiration level affect each other. The level of aspiration is also affected by social values. The feeling of success or failure presupposes proper relations between working conditions, the level of aspiration, and the objective difficulty of the task. Objective success is felt as a subjective success only if the task was accomplished by own forces and with effort [HACK81]. It may be assumed that the main feeling is gratitude, or even envy or hate instead of success, if the task was fulfilled by others.

Lewin et al. [LDFS44] point out the difficulties of direct measurement of the level of aspiration. It is not reasonable to ask after the performance, because after failure verbal expressions might easily be rationalization. Neither does the feeling of failure nor success depend on the absolute level of achievement. In their analysis Lewin et al. mention several factors determining the level of aspiration and the reaction to achievement, e.g. temporary situational factors (success or failure in a series of similar performances, or the transfer of the effect of a previous task of a different kind), general cultural factors like the standards of different groups, the effects of socio-ecomomic background, habitual success and failure; also the reality levels of the subjects.

Lyytinen and Hirschheim [LYHI87], and Lyytinen [LYYT88] have recently introduced the concept of expectation failure which is shortly defined as the inability of the information system to meet the expectations of a stakeholder group. The notion of expectation failure is apparently in good accordance with the aspiration level construct of Lewin et al. [LDFS44], although it is broader in scope because the expectant may not be an actor. Expectation failure depends strongly on the pre-expectations of the stakeholders, or on the aspiration level, as Lewin would say about the actors. As Lyytinen [LYYT88] points out, unrealistically low expectations may seemingly lead to a success while on more objective grounds there should be a failure.

In the analysis of the setting of the level of aspiration a parallel from industrial quality control is useful. There exist two philosophically different approaches. Firstly, we can accept a certain level of errors, and this is known as the "Acceptable Quality Level" -approach. The total cost of quality - the sum of the cost of failure and the cost of failure prevention and appraisal - assumes, according to the traditional thinking, an U-shaped curve. The lowest point of the U-curve determines the optimal amount of errors, and the production process should be trimmed to operate as near as possible to this AQL-point. The second approach considers the cost of errors so high (direct and indirect costs through loss of goodwill, deteriorated working morale, ...) that only faultless operation is acceptable, and errors should be detected as early as possible and corrected immediately. This is known as the "Zero Defect" -approach [BUEL83], and it is the main philosophy behind the total quality control and quality circles, or Z-type organizations [OUCH81], "an organization of continuous improvement" [LILL88]. In this approach there is no fixed standard of performance, but the desired level is constructed by a social process to be as high as possible.

If the variables describing the IS and its impact have been chosen and appropriate levels of aspiration towards them defined, there is still another problem: how the referent point should be set? Dicson et al. [DIWW88] and Davis et Hamann [DAHA88] mention several possible, like comparative -- relative to competitor, normative -- relative to a theoretical idea, goal-centered -- relative to a stated goal, improvement -- relative to past performance, trait -- relative to effective traits. Picture 1 clarifies the situation [HHMM88]. We can make comparisons between several states: (1) the starting situation, (2) the situation that would exist if the project had not been started, (3) the stated goal situation(s), (4) the current situation, and (5) some external standard (the situations in comparable organizations or with comparable IS or peer/expert evaluation). The organization has evolved during the time taken by the project from the start situation to the current one, and these two situations are the ones that have existed in the real world (in addition to the external ones, of course). Comparisons with non-existent states presuppose speculation, may be by writing scenarios, or by performing business simulation, as suggested by Hawgood and Land [HALA88].



Picture 1. The framework for the assessment of the effects of IS development and use. Organization has evolved from state (1) to state (4); the states are described with variables chosen according relevance and contingencies.

The implementation of an IS means an innovation in the organization, and factors such as the complexity, radicalness, originality and divisibility of the system affect the probability of success [IIVA86]. The perceptions of the participants are also obviously affected by these factors. The "implementability" of the proposed IS is taken into account in the initial situation when choosing the actions to be taken, and it affects the ex post analysis through the conceived difficulty of the process. In the starting situation (1) in figure 1 the achievability of the goal state (3) is known only vaguely (or it is a "moving target"), and it is not reasonable to evaluate the success or failure only by comparison of (3) and (4), because it may appear that the original (3) was misplaced, i.e. the goal of the process was wrong, and failure in achieving it is essentially a success. Pressman and Wildavsky [PRWI79, chapter 5] use the great number of clearance points - decisions by different actors - that must be passed as an explanation for the delays (mirroring a part of the hard implementability of the IS) encountered in a project.

4.2 *Goals of the Participating Actors*

The goals of the **central administration** of the University towards the development of SISUH were established by a working committee presided over by the Director of Administration and consisting of officials from administration and two faculties prior to the start of DIRE [DIRE88]. The main goal is to improve the coverage, timeliness and correctness of the registration of the academic progress, and this should be achieved in such a way that there is no increase in the amount of work done by the teachers. The implementation of the department level information system is seen as a means to this end. There are other obvious means for better coverage, of course, such as increasing control and organizational pressure against departments, but they are only briefly discussed here. The amount of manual work should be decreased. The system should be technically flexible in order to accommodate to changes in the curriculum. The system should be secure and cost-effective. A "second order" goal of EDP professionals during DIRE was to find out the contingencies under which it would be profitable to use EDP in departments when processing student information, and what conditions its use presupposes. These facts will be needed later when we proceed with the second set of departments in 1989.

The official goals of the **departments** are consistent with the goals of the central administration of the University as expressed in the project documentation by the departments. Many departments also strive for the development of the skills of their personnel. From the point of view of the DIRE project the heavy utilization of the SIND or STURE in departments can also be considered as a success, as is the removal of the manual files the maintenence of which resulted in extra work. One department wanted to get rid of the study book and the marking of academic progress in it.

The **direct users** of STURE and SIND consider it most essential that the system facilitate the practical operations when manipulating student data. The most important factor is the amount of working time needed, but the more intanglible feeling of the excellent supervision of information concerning students must also be taken into account as a goal. The goals of the **teachers** of the departments are very straightforward: they should not be bothered with the technical processing of student information, someone else should do the job.

The desired technical characteristics of the system were evaluated through the questionnaire by asking the respondents to divide 100 points between several features mentioned in the questionnaire; the respondents could also mention their own characteristics and give points to them, but only few new ones (mainly stressing the importance of useful output) were mentioned by four of the respondents. The averages of the points of the responses (N=24) are:

User-friendliness of the software	15.5
Reliability of hardware	12.1
Reliability of data	19.3
Easyness to change the software	6.1
Good user manual	11.0
Personal instruction	7.0
Rapid functioning of the software	7.3
Well planned back-up procedures	8.4
User courses	2.5

The data problems of SISUH (see next section) have clearly produced the importance of the reliability of data. The points above do not sum up to 100, because some of the respondents omitted this question. All blank items were interpreted as zeroes.

5. The Web Around Development

The data problems with the SISUH - missing or erroneous data concerning academic progress - have been apparent from the late 1970'ies. The generic reason for the data problems of SISUH is the nature of the old system as a means of making statistics. Only a small portion of departments or teachers have had direct results from the system. Students have obtained excerpts from the register only in a sporadic manner. If there have been errors in the register, there has been no easy way for the student to have them corrected. The correct operation of the EDP system processing study attainments has not been necessary in any really vital function of the University. Attitudes toward changing the system of processing academic progress data and thus solving a portion of the problems among the participants of DIRE seem favorable [DIRE89, appendix 2], although general attitudes in the University may be more sceptical.

SISUH has suffered years from a vicious circle: many teachers feel that the system demands unproductive, extra work in sending data to the Actuary's Office. Some study attainments may remain unsent, this leading to poor coverage. Poor coverage, in turn, makes the system inadequate, and the inadequacy felt by its users leads to carelessness, etc. The interpretation of unclear lists of results by the personnel of the Actuary's Office demands extra work and causes delays in data entry, this being seen by the departments as the inability of the system to produce timely statistics. Often the contacts by the personnel of the Actuary's Office to the teachers to correct erroneous data lead to frustration on both sides.

The production lattice [KLIN80, KLIN87, KLSC82], the social and technical arrangements for both the DIRE project and the student information processing within the University can shortly be described as follows.

The Actuary's Office is responsible for the systems planning functions and the maintenance of ACTRE. The Office has a full time planner for the tasks of SISUH, but it is constantly complaining of too great a backlog of work and severe shortage of planning resources. This has manifested itself in forms such as delays in decision making concerning the functions and rules to be implemented into the software modules linking SIND with STURE. The duties of the DIRE project have not been in high priority within the Actuary's Office, and the poor coverage of ACTRE was not mentioned in a recent memorandum [ACTU88] concerning the way to improve the functions of the Office.

The EDP Office is responsible for the management of the project, the writing of the specifications for the software, the education and training of the users, and the provision of help in establishing the department level data bases generated from ACTRE. The resources of the EDP Office, including the financing of the purchase of sotfware from the outside vendor, are not abundant, but at an acceptable level. The software house is quite essential in the production of user-friendly software, because it is not possible to hire a sufficient number of skilled programmers for the EDP Office.

The most important ingredients of the web [KLIN87] are the personal and technical resources of the departments. The persons responsible for the planning tasks of the departments are highly educated, mainly teachers with Ph.D. or Phil.Lic. degrees. They are, however, quite untrained in systems planning, and their motivation for the tasks of DIRE may not be very high, although they express a positive attitude towards EDP and DIRE [DIRE89]. Some of the planners were named to the project by the department head in the first meeting with the project personnel (as a fair surprise to the "planner"), and in some departments there have been several persons in succession because of changes in the department personnel. The attitude of the planners of the departments can be characterized as one of the Ph.D.'s put it: "Somebody must do the job. It was my (bad) luck that I'm the one. However, as I shall be with the department for quite a long time, the job must be done properly. I hope that the tasks are not very time consuming, so that time will be left for research".

The clerical personnel of the department is the key group affecting the failure or success of the DIRE project. If the direct users resist the system or are not capable of using it, problems arise. The attitudes of the users are rather favourable to EDP and DIRE [DIRE89], and some of them are also willing - even demanding the opportunity - to learn new EDP skills. As one of the users expressed, the new skills are advantageous to them because they make those who have mastered them more valuable as workers and thus more competitive in the labour market. This kind of motivation of participation may be a positive feature for the leader of a short-term project, but it may cause troubles later in the form of greater turn-over of the qualified office personnel. Although many of the departments are suffering from the lack of clerks, only one of the participating departments of DIRE has this problem. The proper behaviour of the departments is crucial for the success of the DIRE project and the achieving of the goals of the central administration. This improves the power position of the departments, and they can acquire the technical features of the system as they desire within reasonable resource constraints.

It is obvious that there are no apparent conflicts within the expressed goals of the DIRE project. In this kind of situation counterimplementation [KEEN81, GRLS88] or other complex procedures of resistance toward change are unprobable. The official and expressed goals may give a misleading picture of the organizational environment of DIRE, because there are two generic contradictions in SISUH. Firstly, in the processing of study attainments the teachers from whom the data originates consider the tasks of SISUH as an extra and unproductive duty. During DIRE efforts were made to remove this feature of the old versions of SISUH by making the announcement of the study results to ACTRE as an automated function of the system producing the lists of examination results posted on the departments bulletin boards. Secondly, the traditional nature of SISUH as a means of producing statistics makes it more of a vehicle for supervising the departments than a supportive system for their personnel, and it is obvious [FLEN84] that users' attitudes towards a supervisory system is more negative than it is towards a supportive one. The supervisory features are not removed from SISUH by the DIRE project, on the contrary, but the integration of the system into the functions of the departments may make the supervising more tolerable and less cumbersome.

As a conclusion of this chapter the conditions for the successful implementation of SIND/STURE in a department can be stated to be: (i) usable software produced by the EDP Office and the software house, (ii) hardware and office space for it, preferably PC/AT with a laser printer purchased by the department, (iii) data transmission capabilities arranged by the department with the help of the Computing Centre, (iv) correct or correctable data in the departmental data bases extracted from the ACTRE, (v) training arrangements in the use of the system, including general education in EDP and on-line help by telephone, (vi) time for the users to learn the system, leading to (vii) trained and motivated personnel in departments, obeying (viii) the properly planned procedures to handle the student data in departments, (ix) easy-to-use feedback about data that was sent to ACTRE, and (x) a department head who is constantly interested in the good operation of the system and ready to act if something needs her/his attention. These are necessary conditions (the "clearance points" of [PRWI79]) for the system to be integrated as an institutional part of the functions of the department. A general condition is also (xi) the proper definition of those points in a student's academic progress that are to be registered in ACTRE and those that should only be kept in SIND.

6. An Attempt to Operationalize Success

The model of success and failure in section 4.1 is hard to operationalize. In the beginning of the project there is not enough knowledge and understanding for the proper setting of aspiration level, and when the project is finished, evaluation may have distortions because of rationalization. Thus the use of surrogates is often the only possibility. Job satisfaction, user information satisfaction, or the like are quite customary surrogates, but satisfaction may have nothing to do with, say, organizational effectiveness (see [SIMI88], section I.5.1). In every case, we must also evaluate in retrospect the feasibility of the goals and aspiration level which were expressed in the beginning.

It is unfruitful and counterproductive to try to operationalize a concept without sufficient conceptual analysis. The DIRE context, however, seems to be so restricted that at least some well grounded operationalized measures can be defined from the point of view of the central administration of the University. The operationalization of the success of the departments and their workers seems more difficult (e.g. how much the clerical work should have been reduced to imply success), and remains to be done later after more experience has been accumulated about the essence of the change.

The introduction of a new method of student information processing succeeded in five departments, one department is actively trying to start the use of STURE, the faculty office has made experiments in "production use" with STURE and is waiting software improvements, and two departments have shown no sign of real activity. Thus the failure rate at this moment is 2/9, success rate 5/9, and there are two unclear cases. One of the "failure" departments has announced that it is waiting for SIND to be complete, after which it will reconsider how to connect SIND with its own EDP system. As compared to the "general" level of success in the introduction of new products to market (see e.g. [IMMO87], p. 11, who reports according to [BOAH76], [COOP83], [CRAW77], and [HOPK80] that about 40-80 % of new food and drug products are failures and half or even more of the development resources are wasted on failing products), the success figure of DIRE can be considered satisfactory. If the goal had been to introduce the new way to every department with DIRE, then the project obviously failed.

Success and failure rate are quite obvious operationalizations of the success of the DIRE project, but there is still the problem of implementation efficiency [IIVA86]. Is the amount of resources spent in DIRE proportionate to the results? Without a good point of reference, we must evaluate the outcome only according to the goals and estimated resource consumption of the project. The three goals approved by the Senate (see section 2.3) were reached, nearly according to the budget (planned 2,500 working hours, used about 3,000 working hours). The schedule was kept by a couple of the departments. There were also delays of several months even with the successful departments, with a clear connection in evidence with the failure to pass some of the clearance points (i)-(xi) of section 5.

Was the concrete outcome within my aspiration level? After considerable afterthought and intraspection, I have no clear answer. The goals of DIRE were very modest in the form I wrote them in the project plan, and this may be a symptom of a fear of a failure - still another in the series with SISUH. Perhaps the whole process can best be seen as an intervention that produces testable hypotheses [ARPS87] to be used with the next set of departments, or as a way of organizational learning [ARSC79] to find out the contingencies of effective decentralization of student information processing to a part of the departments of the University.

For the "hierarchic" side of the operationalization of the success two measures emerged: (1) the actual coverage and correctness of ACTRE as measured by the number of complaints students make after they have received register excerpts by mail, and (2) the delay between the date of an examination and the date when results are registrated in ACTRE according to the department or teacher concerned. There are practical problems in implementing the first one, while the second one can be obtained directly from the data base. The use of these measures presupposes much consideration, however, and they will probably only be really introduced after a year or two.

7. Concluding Remarks

Many features of DIRE and the development of SISUH deserve reflection. I shall discuss some of them in this closing section by presenting a couple of questions although I cannot answer them satisfactorily in this paper. The current essence of SISUH is a means of monitoring the formal learning (expressed in academic progress measured in study weeks) of the students. The current system of economic study aid run by the State Centre for Educational Grants and Loans would not work at the University of Helsinki without an EDP system. It is beyond the scope of this paper to speculate on the effect on student culture of a tight economic monitoring according to academic progress, but if we accept this requirement of the Ministry of Education as legitimate, the control function of SISUH towards the departments can be seen to favour the students by helping to produce an efficient account of their academic records. However, does the implementation of SIND mean that the supervising of the teachers is transferred from the central administration to inside the department?

Is the departmental decentralization economically effective throughout the University? The answer is definitively no, if a short term framework is adopted or if all the 124 teaching departments are taken as the users. The sum of work needed in the departments for the implementation and use of SIND and STURE clearly exceeds the amount of work needed if all the direct user functions were to be centralized. Many departments have problems in acquiring competent clerks, and it may not be very profitable to put the extra burden of technical registration of academic progress on already over-stressed departments. In many circumstances routine administration of this kind is best kept out of the departments, as was sharply expressed by a department head [WIIO88]. However, the decentralization must be seen within a larger context with implications upon the visibility of the system, and it is a means of breaking the vicious circle mentioned in section 5.

In retrospect, I think that DIRE and SIND represent a quite idealistic approach. The design ideal is to produce an easy-to-use, flexible system for a competent clerk who is provided with sufficient education and access to support. The reactions to the usefulness and user-friendliness of the software vary. No comprehensive analysis has been made yet, but the pronouncements of the faculties about the memorandum of DIRE [DIRE89] contained diverse evaluations, some saying that the software is rigid, others saying that it functions well. The reactions of the direct users ("the experienced meaning of the system") vary from apparent satisfaction to resignation from the department resulting partly from the use of SIND. Whether the design ideal is realistic and economically feasible remains to be seen. The faculties also approved continuing the work in order to enhance SISUH according to the principles suggested by the DIRE project, and thus the political support for the work is preserved.

ACKNOWLEDGEMENTS

The cooperation of the participants of DIRE project and the personnel of the Offices of the Rector is gratefully acknowledged. The work was partially supported by the Ministry of Education through the doctoral program in information systems. Special thanks are due to Professor Hannu Erkiö because of his comments on the earlier draft of the paper. The language was improved by Eugene Holman.

REFERENCES

ACTU88 Aktuaarintoimiston kehittämistyöryhmän raportti (Report of the Development Group of the Actuary's Office, internal unprinted memorandum), University of Helsinki, 25.11.1988

ARPS87 Argyris, C., Putnam, R. & Smith, D.: Action Science, Jossey Bass, 1987

ARSC78 Argyris, C. & Schön, D.: Organizational Learning, A Theory of Action Perspective, Addison-Wesley, 1987

BJDA88 Bjørn-Andersen, N. & Davis, G. B. (eds.): Information Systems Assessment: Issues and Challenges, Proceedings of the IFIP WG 8.2 Working Conference on Information Systems Assessment, North-Holland, 1988

BOAH76 Booz, Allen & Hamilton, Inc.: A Program for New Product Evolution, in Rothberg, R. (ed.): Corporate Strategy and Product Innovation, The Free Press, 1976

BUEL83 Burril, C. & Ellsworth, L.: Quality Data Processing, the Profit Potential for the 80s, Burril-Ellsworth Associates, 1983

CARL74 Carlson, E. D.: Evaluating the Impact of Information Systems, Management Informatics, vol. 3(1974), no 2

CAME80 Cameron, K.: Critical Questions in Assessing

Organizational Effectiveness, Organizational Dynamics, Autumn, 1980

COCO87 Collins Cobuild English Language Dictionary, William Collins Sons & Co Ltd, Glasgow, 1987

COOP83 Cooper, R. G.: Most New Product Do Succeed, Research Management 26(6), 1983

CRAW77 Crawford, C. M.: Marketing Research and the New Product Failure Rate, Journal of Marketing, 41(2), 1977

DAHA88 Davis, G. B. & Hamann, J. R.: In-Context Information Systems Assessment: A Proposal and an Evaluation, in BJDA88

DASR88 Davis, J. G. & Srinivasan, A.: Incorporating User Diversity into Information Systems Assessment, in BJDA88

DIRE88 Hajautettu opintosuoritusten käsittely, rekisterityö-ryhmän muistio (Distributed Processing of Study Attainments, Memorandum of the Register Working Group), printed in the series "Toimikuntien mietintöjä ja selvityksiä", no 37, Rector's Office, University of Helsinki, 1988

DIRE89 Hajautettu opintosuoritusten rekisteröinti, HARE-projektin loppuraportti (Distributed Registration of Study Attainments, the Final Report of the DIRE Project), printed in the series "Toimikuntien mietintöjä ja selvityksiä", no 39, Rector's Office, University of Helsinki, 1989

DIWW88 Dickson, G. W., Wells, C. E. & Wilkes, R. B.: Toward A Derived Set of Measures for Assessing IS Organizations, in BJDA88

FLEN84 Flensburg, P.: Two Research Methodologies for Studying User Development of Data Systems, in Mumford et al. (eds.): Research Methodologies in Information Systems, North-Holland, 1985

GRLS88 Grover, V., Lederer, A.L. & Sabherwal, R.: Recognizing the Politics of MIS, Information & Management, vol. 14(1988), pp. 145-156

HACK81 Hacker, W.: Yleinen työpsykologia, in Finnish, (original edition: Allgemeine Arbeits- und Ingenieur-psychologie, Berlin, DDR, 1980), Weilin+Göös, Espoo, 1981

HALA88 Hawgood, J., Land, F.: A Multivalent Approach to Information Systems Assessment, in BJDA88

HHMM88 Haglund, H., Heiskanen, A., Manner, K., Mäntylä, H. & Nissinen, T.: Tietotekniikan vaikutukset kunnallishallinnossa (The Effects of EDP in Municipal Administration), The Municipal EDP Delegation of Finland & The Development Center Data Technology in Finland, Helsinki 1988

HHMN88 Heiskanen, A., Haglund, H., Mäntylä, H. & Nissinen, T.: Impacts of Information Systems in Finnish Municipalities: Results from Case Studies, in Kaasbøll (ed.): Proc. of the 11th Scandinavian Information Systems Research Seminar, University of Oslo, 1988

HIRS83 Hirschheim, R. A.: Assessing Participative Systems Design: Some Conclusions from an Exploratory Study, Information & Management, vol. 6(1983), no 6

HISM88 Hirschheim, R. & Smithson, S.: A Critical Analysis of Information Systems Evaluation, in BJDA88

HOPK80 Hopkins, D. S.: New-product Winners and Losers, Conference Board Report No 773, 1980

IIVA85 Iivari, J.: A Planning Theory Perspective on Information Systems Implementation, Proceedings of the Sixth International Conference on Information Systems, edited by L. Gallegos, R. Welke & J. Wetherbe, Indianapolis, December 16-18, 1985

IIVA86 Iivari, J.: An Innovation Research Perspective on Information System Implementation, International Journal of Information Management (1986), No 6, p. 123-144

IMMO87 Immonen H.: Elintarviketeollisuusyritysten tuote-kehityshankkeiden onnistumiseen tai epäonnistumiseen vaikuttavat tekijät (The Factors Influecing the Success or Failure of Product Development Projects within Food Industry), unprinted licentiate thesis, EKT-Series 740, Department of Food Chemistry and Technology, University of Helsinki, 1987

KEEN81 Keen, P. G. W.: Information Systems and Organizational Change, Communications of the ACM, vol. 24(1981), no 1

KLIN80 Kling, R.: Social Analyses of Computing: Theoretical Perspectives in Recent Empirical Research, Computing Surveys, vol. 12(1980), no 1(March)

KLIN87 Kling, R.: Defining the Boundaries of Computing Accross Complex Organizations, in Boland, R. J. & Hirschheim, R. A. (Eds.): Critical Issues in Information Systems Research, Wiley, 1987

KLSC82 Kling, R. & Scacci, W.: The Web of Computing: Computer Technology as Social Organization, Advances in Computers, vol. 21(1982), pp. 2-90

KWZM87 Kwon, T. H. & Zmud, R. W.: Unifying the Fragmented Models of Information Systems Implementation, in Boland, R. J. & Hirschheim, R. A. (Eds.) Critical Issues in Information Systems Research, Wiley, 1987

LDFS44 Lewin, K., Dembo, T, Festinger, L. & Sears, P.: Level of Aspiration, in J. M. V. Hunt (ed.): Personality and Behavior Disorders, Ronald Press, New York, 1944

LILL88 Lillrank, P.: Organization for Continuous Improvement, Doctoral Dissertation, University of Helsinki, 1988

LUCA81 Lucas, H. C.: Implementation, The Key to Successful Information Systems, Columbia University Press, New York, 1981

LYHI87 Lyytinen, K. & Hirschheim, R.: Information Systems Failures - A Survey and Classification of the Empirical Literature, Oxford Surveys in Information Technology, vol. 4(1987), pp. 257-309

LYYT88 Lyytinen, K.: Expectation Failure Concept and Systems Analysts' View of Information Systems Failures: Results of an Exploratory Study, Information & Management, vol. 14(1988), no 1

NIEM89 Nieminen, M.: Työtyytyväisyys ja tietotekniikka (Job Satisfaction and Information Technology), unprinted master's thesis (with honors), University of Helsinki, Department of Political Science, Helsinki, 1989

OUCH81	Ouchi, W.: Theory Z, How American Business Can Meet the Japanese Challenge, Addison-Wesley, 1981
PRWI79	Pressman, J. L. & Wildavsky, A.: Implementation, Second Edition, University of California Press, 1979
RAHU88	Ranta, J. & Huuhtanen, P.: Informaatiotekniikka ja työympäristö, osa V, informaatiotekniikka pankki- ja vakuutustoiminnassa, in Finnish (Information Technology and Working Environment, Part V, Information Technology in Banking and Insurance), The Finnish Work Environment Fund Publications n:o A5, Espoo, 1988
SASÄ89	Saarinen, T. & Sääksjärvi, M.: Tietojärjestelmien onnistuminen suomalaisissa suuryrityksissä (The Success of Information Systems in Large Finnish Enterprises), Helsinki Business School, Report D-116, 1989
SCHÖ83	Schön, D.: The Reflective Practitioner, How Professionals Think in Action, Basic Books, 1983
SIMI88	Similä, J.: Modelling and Analyzing Empirically the Success of ADP Systems Use, Doctoral Dissertation, Institute of Information Processing Science, University of Oulu, Oulu, 1988
SIMM77	Simmonds, C.: The Nature of Future Problems, in Linstone, H. & Simmonds, C. (Eds.): Futures Research, New Directions, Addison-Wesley, 1977
SUEV78	Susman, G. I. & Evered, R. D.: An Assessment of the Scientific Merits of Action Research, Administrative Science Quarterly, vol. 23, no 4, 1978
WIIO88	Wiio, O.: Eriävä mielipide konsistorille annettavasta rekisteröintiä koskevasta lausunnosta (Dissenting Opinion about the Pronouncement to be Given by the Faculty of Social Sciences to the Senate about Registration of Student Academic Progress; the pronouncement was asked about a draft of [DIRE88]), unprinted memorandum, Department of Communications, University of Helsinki, 6.2.1988

The STEPS workshop - Christiane Floyd discusses with the participants

Paper submitted to the 12th IRIS Conference, Skagen Denmark.

CREATIVITY IN A SYSTEMS DEVELOPMENT STUDY PROGRAM

Stig C Holmberg
Informatics and Systems Science
University of Östersund, P.O 373
S-831 25 ÖSTERSUND, Sweden

tel. (46) 63155388. 63105711
Email: STIHOLÉSEUMDC51

Abstract:

Based on ten years of teaching in information systems development a discussion is given around systems development and creativity. Starting with three basic questions concerning creativity and systems development the need for creativity in information systems development is argued.

In the second section we give a pragmatic definition of creativity in systems development and also discuss around some related concepts and theories.

In a third part of the paper we will review and discuss the theoretical material which we have found to be a prerequisite for creative thinking in systemeering. Our arguments for the students being oriented about these theories are given.

In the next section we discuss around practical matters, how to apply, stimulate and impose creativity in the systemeering courses. Here also some examples of course organization and formulation of project works to give training in creativity are demonstrated.

The paper is closed with an evaluation of our results so far and with some ideas and plans for the future.

Key words: Creativity, Information Systems development, Systems Design, Curriculum.

1 Background

Creativity is stated as one of the main goals for the systems development program at the University of Östersund (SVE2OOO, 1987). However a plan is a plan and reality is reality and never they meet. In other words, we have an uneasy feeling that the students coming from the university in our little town are not any more creative than those leaving other universities. Also, there is no indication that they are any more creative then leaving the program then upon entering.

Anyhow the last ten years of trying to implement creativity have evoked a lot of experience and ideas. There are also a lot of actual papers and books around, which are discussing different aspects of creativity. In the rest of this paper I will then discuss some of those ideas, which may finally be synthesized to form the base for a renewed and hopefully more successful attack upon creativity.

For the purists at last, it must be stated that this paper is more a work of ideas rather than of scholarship.

2 Creativity in Systems Design and Systems Development

At least three basic questions seem to arise then we start to discuss creativity. First, is there anything in the human mental capability which can be labeled as creativity, and in that case, what would a proper definition look like? Second, is creativity according to our definition a valuable or maybe even an unmissable property for system developers? Last, is creativity something mysterious left to chance and only found as a property of a genius or is it something that can be identified and learned, at least to some part, by everybody ?

There is a great amount of literature, both scientific and opportune, dealing with creativity. It is not possible to discuss all this literature in this paper. I just want to mention Vernon (Vernon 1970) and Sternberg (Sternberg 1988) as possible entry points. As a consequence of not going deeper into the literature I also will give the answers to those questions in the form of assumptions rather then in the form of scientifically backed conclusions.

First we assume that creativity is a part of the human mental repertoire and that it is closely related to design, inquiry and problem solving. Taking Warfield's definition of design (WAR 88) creativity in this way will include a rather broad set of human abilities. We have among others the skill in seeing and formulating problems together with the faculty to find improve, evaluate and implement new ideas. On one side we have the ability to find original and new combinations and new aspects, on the other side we also have the power to implement the decided solution.

We also assume that creativity is a highly needed skill among system developers. The world is facing severe threats in many areas and the solutions obviously are not to apply "more of the same" in a stereotype way. Even with an engineering approach of ours it is worth observing that the word 'engineer' has the same root as genius (compare the 'genie logiciel' in French with the English 'software engineering'). To work with an engineering approach obviously oblige you to strive for genius and creativity.

Also in the last point we have a positive assumption. Everyone can improve his/her power of creativity. Here it is not interesting to compare with others, everyone can perhaps not become a new da Vinci, but anyone can start to indicate a beginning development toward higher creativity after just a short time of training.

3 A Theory of Creativity in the context of Systems Design

In this discussion we will restrict the concept of Creativity to activities related to our educational goals, i.e. design and development of information systems and to problem solving in complex system contexts. Within this sphere creativity will be seen as a structured approach for developing plenty of new and useful ideas applied to dissolving issues of relevance to organizations and individuals. In psychology literature (e.g. Reber, 1985) uniqueness and novelty are put as criteria of creativity. Those criteria will be relevant also in our use of the concept.

Another way to look at creativity is to switch the interest from the product or final result to the creative process itself. In this way we will find very interesting connections with a set of related concepts from both Systems Science Theory and Psychology.

We have here for example (Varela, 1984) the studies of relations that start processes which are independent of their embodiment. According to this view creativity can be seen as the event then a changing system moves into a new and novel state (Peeno, 1989). Systems characterized by equilibrium, homeostasis and adjustment will not be open to phenomena of change. If we want change and creativity to happen we have to look for or design processes which are open, dynamic and evolutionary (Bertalanffy, 1968).

In the traditional mechanistic world view of Newton creativity is mostly seen as a process that is rational and which has laws and antecedent conditions. The results of the process hence will be theoretically predictable. Change here occurs in a world governed by unchanging laws. An opposite view, is that creativity is a process that transcendents rational principles. This view assumes that the creative person acts in a way that is not the result of what is present in his experience before he originates the novel product. It is also assumed that while he is engaged in the act of creation he is not in full control of himself (Hausman, 1984). This conception of creativity is very congruent with the theory of change implied in the idea of dissipative structures (Prigogine, Stengers, 1982). When fluctuations have forced a system into a far-from-equilibrium condition it will come to a bifurcation point. A point there it is impossible to determine in advance the next state of the system. Here changes will occur in a world in which the laws themselves undergo changes.

According to the Living Systems Theory (Miller, 1978) the mind can be taken as an open, abstracted living system, in which creativity is the dynamic, evolutionary process by which new states or systems emerge. In the light of this evolutionary paradigm the creativity process can be understood as a process in which there are bundles of trajectories or a divergence property.

Then, how does a new mental process arise from within the old habits and states of mind? Miller assigns this transformational function to the "associator". The subsystem which would be incapable of creativity were it not for self organization and emergence.

In short, comparing the creative process with a nonequilibrium theory of evolution may enable us to broaden our understanding of the cognitive activity that occurs during the creative state (Preeno, Merker, 1989).

4 Implementation

Regarding the implementation of creativity into the study program I think that it must have both a theoretical and a practical side. The theoretical knowledge may provide the basic insight of learning and thinking processes and the nature of design and problem solving. But in the same way that you can't learn to swim without water you can't learn creativity without suitable and sustained practical training.

In subsection 4.1 I will provide a compilation of ideas that I think could help in establishing the mental base for creativity. The list is open ended, more points can be added. At the same time it may be possible to find ideas of a higher quality than those in the list. References which then ought to replace those given in this first version.

After that, subsection 4.2 will provide some practical guidelines about how to provide the students with the necessary creativity training during their time at the university.

4.1 Creativity in theory

First I think that it is important to make the students aware of the ongoing discussion about the role of the universities and different ontological schools. The university is not necessary a job training machine and learning do not need to be equal to memorizing "objective" facts. Ivanov in this context has written some very interesting reports which put this discussion into a systems and an information processing perspective (Ivanov ,1980a and 1980b).

4.1.1 The theoretical base

As a second point, in the "classical" Systems Science literature I have found a whole set of ideas which can help in opening up the mind for creative learning and creative thinking (creative processes).

Miller (Miller 1978) with his Living Systems Theory may provide the basic understanding of the structures and processes producing creativity. It may be enlightening to understand that creativity is not so much a product of memory but more of another subsystem called associator, not so much of the statical structure but more of the process (the flow of information through the system).

Creativity, from another aspect, is not merely a question of problem solving but as much a question of problem seeing. Van Gigch (V Gigch 1978) in this context gives an interesting discussion about the influence of world view (Weltanschauung), premises, assumptions and cognitive styles upon decision making and design. His discussion of "truth" may also help to loosen up any mental restrictions.

An awareness of different Inquiring Systems (Churchman 1971) may also be of crucial value. Most of our students seem to be 'lockeans' and, even worse, totally unaware of the possibility of being anything else. On the other hand a Singerian attitude may be more prosperous for creativity.

In order to work creatively it also seems necessary to have a good understanding of the concepts underlaying the machine age on one side, i.e. analysis, reductionism, determinism and mechanism and the systems age and systems thinking on the other. Here Ackoff (Ackoff 1981) gives a good summary together with clearing discussions of development and problem solving.

The ideas of Prigogine (Prigogine 1984) about dissipative structures and bifurcation points also seem to be very powerful for our understanding of creativity.

One other conception of creativity is closely related to complexity. A creative act is the creation of a structure which has a higher complexity then the structures we had before. Stafford Beer (Beer 1979) can here give us a better understanding of complexity. Beer's elaboration of his system four also shows that creativity is not restricted to individuals. Creativity can also be found in organizations and on other system levels. An other observation here may be that creativity has to be supported by some sort of infra structure (Holmberg 1988).

Yet another piece of understanding is given by Warfield (Warfield 1988). His new cosmology, his design laws and design principles together with the distinction between descriptive sciences and design sciences provide many good concepts for a creative work. The importance of dimensionality (Warfield 1987) is another major contribution in this area.

Boulding at last (Boulding 1978) provides a very deep discussion around evolution, dynamics and evolutionary change.

4.1.2 Some actualities

Creativity has during the last years been a main topic within the International Society for Systems Science and a lot of recent papers can provide many powerful and enlightening ideas (Ledington 1989).

The word "system" is often and perhaps in an unreflecting way used in systemeering and systems development. The concept is discussed by Checkland (Checkland 1988) and his conclusion is that system is the name of an abstract concept to be used in a conscious process of trying to understand (Learn) the world. "It is an error to use the same word for an abstract epistemologycal device and for assumed ontological entities in perceived reality". But he also finds that the systems concept used in the right way can improve the learning process. Similar thoughts are expressed by Jäderlund (SYNT 1988).

Davies and Ledington (Davis 1988) discuss creativity within the Soft System Methodology (SSM). In order to stimulate learning and creativity to break out of mind-sets and other restrictions they recommend that the real world under concern ought to be transformed into a metaphorical model.

Peeno and Merker (Peeno 1989) demonstrate how creativity can be seen in an open systems perspective. Drawing from research of Barron, Miller, Laszlo and Prigogine creativity is here elucidated in terms of process, chance and instability, bifurcation, open, dynamics and evolutionary, The emergence of truly new things and third state systems.

The isomorphism between creative persons and creative processes on one side and self-organizing dissipative structures on the other is further elaborated by Montuori (Montuori 1989). The mind of a creative personality can be considered as an open system. A personality which is perceptual rather than judging. "Subjects with a preference for simple order attempt to maintain an equilibrium which depends essentially upon exclusion, ... refusing to see parts of reality that cannot be assimilated so some preconceived system". "Creative persons, on the other hand, favor disorder and complexity, but only because they wish to integrate it into a higher order synthesis".

4.1.2 Complementary sources

Probably it is necessary to motivate the students to try to be creative and developing. Both Nevitt (Nevitt 1980) and Banathy (1989) provide good evidence for the need of creativity in challenging the great and systemic problems in front of us.

Wallin (SYNT 1989) speaks about potency and actuality. The potency in front of us represents an infinite set of possibilities. It is then up to your creative acts what part of this potency you want to realize into actuality. With his monitoring system Wallin also shows how creativity can be put in the center of our every day actions, on an operative, tactical and strategic level.

For the insight into creative proactive and creative awareness we have found Folkeson (Folkeson 1981) to be a very good source.

We also think that it may be valuable to see what people outside our own field think about creativity. Here Rowe (Rowe 1987) from the field of architecture may be a good example. The close connection between inquiry, design and creativity in this work is specially striking and interesting.

4.2 Practice.

Our own experience at least tells us, what we do not ought to do in order to make our students more creative. It is definitely not enough to speak about creativity at one occasion and to let the students perform some standard creativity exercises at that one shot show. The direct effect will normally be very good but the lasting effect will always equal zero. A general conclusion may then be that creativity training must be a continuous activity which traverses all aspects of the curriculum.

Another exercise which has given a bit better result is to let the students write a "creativity diary" during some months. This means that the students are asked to write down not only all their creative ideas but also the relevant conditions around the idea finding. Relevant conditions may be time, place, activity, mood, type of problem and the period under which the problem had been considered. To observe our own creativity seems to make us more self aware of the conditions which are favorable for our creativity and finally we will be able to reinforce those conditions. We think that these exercises can be developed further and that the results can be utilized in a more systematic way.

All courses must be organized in order to give room for creativity. One way to achieve this is to make all training, group works and examples applied on real world problems without preestablished true solutions. The courses ought to be organized into rather big blocks of 20 weeks. The goals for a block may be set rather precisely but there must be place for a great flexibility within the block on how to arrive at the goals. In some areas it is more or less necessary to provide the students with some rather accurate defined facts, laws, syntactic rules and algorithms. There is not much creativity related to those activities but in this cases it is important to give time for both the strict acquisition of facts and the free and creative application of the facts on real world problems.

All application training must be concentrated on sharping thinking skills and not on traditional training of memorizing abilities.

There are indications that our time with an emergence of a systems age will call for radically new studying forms. It is not possible to define those forms today but it must be an ongoing and creative activity to approach those new forms as fast as possible.

We have also found it difficult to teach methodologies. Whatever you as a teacher may say about using the method as a guideline and to adapt it to the situation. The students tend to follow it word per word, even if they find it very stupid.

In order to give room for creativity we seem to be obliged to teach methodology without referring to any specific method. On a general level we must be very observant of any creativity killers in the curriculum.

Probably we also have to pay attention not only to individual creativity but also to group creativity. To arrange situations and conditions which will start and maintain creative processes in groups will be a main concern for the teachers.

Design studios (Warfield ,1988), (Holmberg ,1988) could be a physical help in the environment to stimulate creativity. It will hopefully be one of our high priority goals to establish such studios and to have our students working in them.

According to Thorsheim (Thorsheim 1987) we also have to pay greater attention to each individual student in order to free his/her full potentials. As teachers we must spend more time to know each student individually and to understand the motives and hopes behind their studies. The teacher and the student will be able to work together in a more positive way.

Allain (Allain 1989) argues that creativity and development takes place at the boundary between different cultures. In line with this observation, foreign teachers ought to teach at each university (in their own language) and each student must take for example at least 25% of the courses for a foreign teacher. Can we perhaps start within the IRIS-community?

Hill (Hill 1989) argues for the importance to develop the right inner environment within each student. I can not judge whether the relaxing exercises advocated by Hill are the right way to go, but I find the establishment of a creative climate, both inner and outer, as very important. The conclusion anyhow is that all the teachers involved in the program ought to start thinking and discussing around creativity, and if possible they ought to develop a common strategy to improve the creative climate of the program. The transferring and shaping model for teaching, if present, ought in most cases to be replaced with a traveling or growing theory for teaching (Fox, 1983).

5 Conclusion.

First, if we want to stick to our creativity goal, we have to reorganize our course material and improve the study environment we are providing our students.

Second from the survey in chapter three we draw the conclusion that there is a very rich source of knowledge and experience to utilize for those wanting to sharp the creativity edge in their educational programs.

The main point will then be a question of filtering and organizing the material. What we have said in chapter four can in this context be seen as a first tentative design of a creativity supporting study program. Future measurements and assessments will then show to what extent the design need to be improved and tuned.

The last question will then be a question of measuring creativity and changes in creativity in order to evaluate our program. Measuring creativity surely is not an easy task but Richards (Richards 1989) has shown a method to conduct such quantification and measurement. In applying those questionnaires and measurements recommended by Richards we hope to find some of the significant conditions behind creativity and at the same time it will be possible to refine the assessment and measuring methods.

6 References

Ackoff R. (1981); Creating the Corporate Future. J Wiley & Sons, New York.

Allain M. (1989); Notre langue face à L'Europe (French). Le Monde, No 13827, jeudi 13 juillet, Paris.

Barron F (1988); Putting creativity to work. In The nature of creativity, ed. Sternberg, Cambridge University Press, Cambridge.

Beer S. (1979); The Heart of Enterprise. Wiley, Chickester.

Boulding K. (1978); Ecodynamics. Sage Publications, Beverly Hills

Checkland P. (1988); Images of systems and Systems Image. In Journal of Applied Systems Analysis, Vol 15, pp 37-42.

Churchman C.W. (1971); The Design of Inquiring Systems. Basic Books, New York.

Davies L., Ledington P. (1988); Creativity and Metaphor in Soft Systems Methodology. In Journal of Applied Systems Analysis, Vol 15, pp31 - 36.

Folkeson A. (1981); Kreativ Problemlösning. (Swedish). KTH, maskinelement, Stockholm.

Fox D. (1983); Personal theories of teaching. Studies in Higher Education 8(2).

vGigch J (1978); Applied General Systems Theory. Harper & Row, New York.

Hausman C. (1984); A discourse on novelty and creation. State University of New York Press, Albany.

Hill R.C. (1989); Freeing the Internal Environment for Problem Solving and Creativity. A precursor to Education and the Future. In proc of the 33rd annual meeting of the International Society for Systems Science, Edinburgh.

Holmberg S.C.(1988); Designing and Planning Facilities in GeoInformatic Systems. Report No 13, Geodesy, Royal Institute of Technology, Stockholm.

Holmberg S.C. (1989); Designing a systems Science Program. Paper submitted to the 33rd Annual Meeting of the International Society for Systems Sciences, Edinburgh.

Ivanov K. (1980a); Forskningsanknytning av universitetens grundutbildning (Swedish). Universitetet i Linköping, matematiska institutionen, Linköping.

Ivanov K. (1980b); Systemvetenskap och fragmentering av kunskap (Swedish). Universitetet i Linköping, matematiska institutionen, Linköping.

Ledington P. (Ed) (1989); Proceedings of the 33rd Annual Meeting of the International Society for Systems Sciences, Edinburgh.

Miller J. (1978); Living Systems. Mc Grow Hill, New York.

Montuori A. (1989); Creative Human Systems.Paper submitted to the 33rd Annual Meeting of the International Society for Systems Sciences, Edinburgh

Nevitt B. (1980); ABC of Prophecy. Canadian Futures , Toronto.

Prigogine I., Stengers I. (1982); La nouvelle alliance. Editions Gallimard, Paris. ((1984); Order out of Chaos. Bantam Books, New York.)

Peeno L. Merker S. (1989); Creativity, An Open Systems perspective. In proc of the 33rd annual meeting of the International Society for Systems Sciences, Edinburgh.

Reber A. (1985); The Penguin Directory of Psychology. Penguin Books, New York.

Richards T., Puccio G. (1989); Creative, Problem- Solving and General Systems Theories. Paper submitted to the 33rd Annual Meeting of the International Society for Systems Sciences, Edinburgh

Rogers C. (1969); Freedom to Learn. C. Merril Publishing Company, Columbus.

Rowe P. (1987); Design Thinking. MIT Press, Cambridge.

SVE2000 (Swedish) (1987); Report, University of Östersund, Informatics and Systems Science, Östersund.

Sternberg R. (Ed) (1988); The Nature of Creativity. University Press, Cambridge.

Thorsheim H., Roberts B. (1987); Growing students "at their Scheme edges": Empowering learning through mentoring-partnerships. Paper presented to the 31st Annual Meeting of the International Society of General Systems Research , Budapest.

Wallin E. (1980); Vardagslivets generativa grammatik (Swedish). Liber Läromedel, Lund.

Warfield J. (1988); Development of Generalized Design Theory and Methodology. George Mason University, Institute for Advanced Study in the Integrative Sciences, Fairfax.

Warfield J. Christiakis A. (1987); Dimensionality, Systems Research, Vol4, No 2.

Vernon P. E. (Ed) (1970); Creativity. Penguin Books, New York.

Initiative in Cooperative Design

Berit Holmqvist & Kim Halskov Madsen
Information Science Department
Aarhus University, Denmark

1. Introduction

The main goal of the research programme "Computer Support in Cooperative Design and Communication" at Aarhus University is to develop advanced computer support for cooperative design and, as a basic, to gain understanding into cooperative work - particularly cooperative design, [Andersen ét al 1987] and [Bødker ét al 1988].

In one of the early projects of the research programme we have used an existing design tool, HyperCard for the Macintosh, to design computer support for academic writing. The project has been conducted together with researchers at "The Centre for Cultural Research" (CCR), Aarhus University. The purpose of the project was twofold, to gain experience concerning the usability of the specific design tool and to increase our understanding of cooperative design. In this paper we report on the latter.

The prevalent conceptions of cooperative work emphasize the common background of the people involved, shared goals, minor hierarchical control and no competition, see for instance [Sørgaard 1987]
Others have emphasized the use of shared tools and materials as opposed to the communicative aspects of cooperative work, see for instance [Johnson & Weawer 1986]. In this paper we focus on communication, not as exchange of information between equals, but as a means for resolution of differences in background and goals.

Our main purpose is to try out discourse analyzes as a method for understanding cooperative work. We will not be able to give an over all description of the project or come up with an evaluation or any recommendations, but we do hope that the paper might generate some ideas for the future work.

To use language is to communicate and to communicate is to cooperate, [Allwood 1976]. This does not mean that communication could not be used to pursue a conflict. On the contrary, we will argue that even in cooperative work situations, there are conflicts. An important part of the cooperation is to handle those conflicts.

Now, in a conversation there is always one of the speakers at the time who has the initiative with respect to the goal and the subject of conversation, and thereby momentarily the control of the conversation. This is the source for conflict that we intend to uncover. Who is taking and keeping the initiative, when and by what means?

Our material is tape recordings of a 2 hours design meeting from the CCR project. A series of meetings was held together with the people from CCR.

At the first three meetings at CCR, people from the centre gave a brief introduction to their daily work and we presented various ideas about how computers could be used in the specific cases. Based on these early meetings, it was decided that we should design a small system for S# together with him. At the fourth, meeting S# who has already been experimenting with HyperCard, presents his ideas about the system. He has actually made an early prototype himself.

The discussion of S#'s ideas continues in meeting number five. R#, one of the designers from our department, takes over and start systematizing S#'s ideas. At the next meeting, number six, R# presents to S# his version of a prototype, and together they try to construct a new one based on the first two. Based on the discussions of this meeting, R# has redesigned the prototype. He presents the prototype to S# and S# try out the prototype himself in a last meeting.

In this paper we concentrate on the 6th meeting.

Since we intend to describe the dynamics of the conversation we are primarily interested in recurrent patterns, which means that we have to look at, and present, a fairly large amount of material. In the paper we present and analyze three longer transcripts from the tape. The method of analysis is presented in the following section.

2. Method of analysis

The method of analysis we use is of a general nature. It can be applied to all types of conversation.

Linguistic cooperation is to negotiate and define the constellation of *roles* in a situation, a shared linguistic code (a common language) and the rules for conversation. The roles, code and rules are partly defined through the participants expectations of each others rights, obligations and skills [Berry 1987]. These expectations are formed by the participants specific knowledge of each other and general knowledge about situation types [Holmqvist & Bøgh Andersen 1986]. This forms the *background* of the situation.

In many situations the participants obligations, rights, knowledge and skills are formally defined. In such a role constellation the rules for conversation and the use of code usually follows a rather institutionalized pattern To illustrate this we use an example from a classroom duologue:

Teacher:	What is a regime? Do you know Per?
Pupil:	Well, it is such ones that rule people.
Teacher:	Louder!
Pupil:	Such ones that are ruling.
Teacher:	Yes, that is one way of putting it. In Afghanistan we have a regime that The Soviet Union more or less placed there. A government. [Free Translation from Einarsson 1981]

The teacher has the right, the obligation and the skill to influence and control the pupils knowledge and social behaviour. Therefore he is the one who has the right to ask questions, and to evaluate and comment on the answers given by the pupil. The pupil has the obligation to learn and show that he has done so. The acceptance of these roles is reflected in the rules for conversation of the dialogue above. Teacher *initiate* the conversation with a *request* of information. Pupil *responds* by answering the question. Teacher *requests* that the pupil behaves properly. Pupil *responds* by obeying. Teacher gives *feedback* by *evaluating* the answer and *commenting* on it [Coulthard 1977]. By the comment, the teacher attempts to *change the code* of the pupil. The pupil does not master the professional language of political science, but part of the teachers obligations is to teach him so, therefore he changes the expression "such ones that are ruling" to the concept "government". The reader probably recognize this and similar situations and remember what happens if you brake the rules.

If the roles are not formally defined, the code, the rules and the roles have to be negotiated and established during the conversation.
In the classroom dialogue, typical the teacher has got the initiative and the control of conversation most of the time. In a situation of less formality the initiative can be taken over by one of the other speakers or be kept by the first one.

In sum, the background is the stable basis for the conversation, which create the participants expectations of which roles to play, which rules to follow and which code to use. The roles has a dialectic relation to background as well as to the rules and the code of the conversation. Our direct access to the situation is through the code and the rules. Below we introduce a framework concerning code and rules.

In the classroom the difference in code are due to difference in having or not having concepts for a certain phenomenon. In other situations the difference in code could simply be a difference in perspective, e.g. the participants master the same code but choose different viewpoints. This is the case in the actual situation. To grasp the differences in *perspective* we characterize the code according to the use of different concepts for the same phenomenon as well as which aspects of the phenomenon you emphasize. Hence, the code related aspects we are looking at are:

CONCEPTS. A phenomenon that you have a kept together image of, realized as one word or a lexical phrase in your language.

EMPHASIS. What you put the spotlight on. There is a differences between saying that "the bottle is half empty" and to say that the "bottle is half full", or between " the bottle is full" and "it is the bottle that is full".

The means by which the one or the other perspective is initiated and kept, is visible in the way the conversational rules are observed or broken. The rule related aspects we are looking at are:

FRAMES. Boundary markers put up to mark when a topic of conversation is ended an a new one starts, realized by words like "well", "right", "now", "good", "okay" etc., strongly stressed. [Coulthard 1977].

FOCUS. A meta-statement that tells what topic of conversation that is intended. For instance, "(Now) let us talk about cats". [Coulthard 1977].

MOOD. Mood reflects the different interpersonal functions of language in the sentences types: imperatives, interrogatives or declaratives. Imperatives signals requests for action, "bring me the cat!", interrogatives requests for information, "where is the cat?" and they are both oriented towards the receiver. Declaratives are oriented towards matter of facts and does not actively involve the receiver "The cat is on the roof". Mood are closely connected to modality.

MODALITY.The term is traditionally used to measure the force of a statement. An indicative can be modified with auxiliaries to indicate if an utterance is statement about a fact: "the cat is on the roof", or about a possibility, "the cat can walk on the roof" or about an expectation, wish or an obligation "the cat should be in the yard". [Halliday 1978] In resemblance to this the force of statement or a request can be modified in a lot of different ways. "There is a beer in the refrigerator". "I believe there is... " "Give me a beer!" "Will you please give me a beer!" Here we use the term in this expanded version.

3. The analysis

Background of the design situation

R#, the designer, and S# , the end user, have a common background. They are formally equals. They are former fellow students and colleagues. Particularly, they both have thorough knowledge about humanistic research work. Due to a similar educational background they are able to understand each others professional languages. They are good friends and have trust in each other. S# has voluntarily entered the project out of pure interest. The computer system R# and S# are designing should be tailored for S# work process. At the same time R# is interested in testing some of his design methods. There is no conflict about this bias. Apparently, the situation is close to ideal cooperation: the participants have common interests and goals and there is neither hierarchical control nor competition.

But R#s job is to teach and do research about design of computer systems. He spends a lot of time programming and he has a general knowledge about design of computer systems that S# has not. R#s experience with the design environment, HyperCard, is limited but more advanced than S#. S# job is to do research within "history of science". His work is mainly to write books and papers. But he also has some knowledge about HyperCard programming.

The project group, in this situation represented by R#, has made some commitments to the research council financing the project, and to CCR, here represented by S# , who is spending time on the project. It is a kind of seller/buyer relation, but since this is a university and not a private enterprise it is more prestige than money that is at steak. R# has initiated the project and made the initial contact to CCR to start with. The people he is dealing with is his former colleagues and teachers. He is the only one in the project group that has this relation to the users and feels responsible for the project. S# on his hand knows he is taking part in a research project and can't act as a sincere byer. He does not want to destroy any hidden research intentions and therefore use a policy of wait-and-see.

How do the participants handle this situation? On the one hand they are two boys playing, on the other hand there should be a reasonable product coming out of the project. A product that satisfies both S# and R#s interests.

The specific situation.
The following analyzes is based on transcripts of the tape recordings. Phenomenon like stress, pitch, smiles etc can't be seen in the text. It should be pointed out though that we both were present at the meeting and that the interpretations also is a product of our direct observations, and that any discourse analyses always implies an interpretation.

R# takes the initiative in the conversation setting the stage:

> Jag har gjort en del av de delssystemer som du inte har gjort och så vill jag visa dig vad jag har gjort, alltså prøva att diskutera det och så prøva att sætta det samman som du har gjort med det jag har gjort.

And the conversation starts according to the intended pattern, with a demonstration phase. After that they do not stick to the intended chronology, but starts putting the to systems together to let the discussion grow out of that. The session ends with a phase of pure hacking that R# had not foreseen. In the following we have picked out three scenes, with longer examples from the conversation process, that illustrates these three phases, and we made running comments to them according to the method of analysis.

Notes on scene 1

R#	Jeg har gjort det, at jeg har lavet nogle af de delsystemer som du ikke har lavet. Jeg kan lige prøve at vise dig hvad det er jeg har lavet, og så kan vi prøve at diskutere det og så kan vi prøve at sætte det som jeg har lavet sammen med det som du har lavet.	R# takes the initiatives by defining the topic. First he is going to show S# what he has done since last session, and then they can discuss it and finally put together the two systems. R# indicates in various ways that he has the initiative. First, he is the one who is acting: "jeg viser" alternatively he could have said "let us look at". Second, he define the ordering topics of the conversation.
S# R# S#	....	Social talk about SYDPOL, a Scandinavian Research programme.
R#	Ok!	The frame "ok" indicates that the social talk now is over and the session starts for real. R# has already declared that he is the one to start.
18		
R# S# R#	Det her, det er emnelisten ja, der har vi emneord liggende	The declarative mood marks that this part of the conversation is a demonstration where S# should not be an active participant. R#'s presentations is oriented towards the system itself. What the display screen shows is in emphasis, not S#'s working process, though words from it are used. "Emneliste" og "emneord" are words that S# earlier have used in describing his working tools.

S# ja
R# Det her, det kaldes for en bogliste, det er muligt at vi skal finde et andet ord til det -- det er altså en liste af referencer til en bestemt

The mood is still declarative. The modality though is weakened ("det er muligt") and hereby opens up for a discussion about possibilities: R# indirectly asks for a suggestion from N#

S# til en bestemt
R# til til bøger
S# ja, udvalgt på - eller bare alle bøger
R# alle bøger - det er simpelthen listen over bare alle bøger
S# ja - ok - ja

S# can't partake in the discussion since he is not sure about the function of "boglisten". So he request complementary information. But he does not use the information in the discussion. He just give the feedback that he is satisfied with the information: ("ja - ok - ja") and R# goes on with the presentation.

R# Det er listen over alle noter --- Ja, så kan jeg lige lige vise - ja så kan man - jeg har lavet det sådan - jeg er ikke sikker på at det skal være sådan- man nu har jeg lavet det sådan at man kan gøre tre ting ved et sådant emne eller sådan et element på sådan en liste der. - Man kan søge i den
S# ja

The concepts used are operations ("pege", "søge") that the you can perform on objects ("boglisten") Not concepts from the working process.

R# og så kan man pege i den og så kan man skrive i den - det der med at pege i den, det er måske det enkleste -- nu tager jeg boglisten der - og pege i den, og det betyder så at hvis du peger på den så kommer du over til det kort som som har det navn der.
S# ja

The systems perspective is further reflected in the Macintosh and HyperCard concepts, for instance: "og det betyder så at hvis du peger på den så kommer du over til det kort som som har det navn der). The mood is still declarative.

R# Og så tænkte jeg på at - de- for eksempel det element eller den beskivelse der står i boglisten kan det simpelthen ikke være den måde man referer til det i en artikel
S# ja
R# - altså "H.C. Andersen 1975", for eksempel.
S# jo

Now R# make's suggestions about how the system can be used. The same unique identification in the system is suggested to be used as identification in papers, books, etc. The identification system in the system is transposed to the products produced by the system. The mood change to interrogative "kan den simpelthen ikke...?" And the modality is weak ("kan"). But the rhetorical figure "simpelthen ikke" indicate that he is not opening up for discussion, it is more like a rhetorical question.

R# Det er passende langt
S# ja'
R# og så identificerer det entydigt og du kan bruge det alligevel.
S# ja

The arguments are there, and the ones which are related to the system are emphasized: "passende langt" and "identificerer det entydigt" , while arguments related to the use of the system "og du kan bruge det alligevel", comes in second hand.

R# altså du kan
S# det behøver ikke at være mere end
R# Du kan kopiere det direkte ind på manuskriptet så
S# ja -- det ku' det godt - jo det er sikkert rigtigt nok - såden --- det har jeg ikke, altså nogen færdig mening om. Det er nok rigtigt.

R# has an idea about how the system may be used but S# is not completely able to follow him, and shows explicitly his insecurity.

R#	Det, det, jah, - du er ihvert fald - uanset - du er nødt til at have nogle kortfattede navne på det der- der var det ihvertfald en idé simpelthen at vedtage at man bruger - den måde man refererer til en bog - bruger man simplethen her.	R# admits that there might be a topic for discussion here by starting: "Det, det jah" but he instantly tries to withdraw "Du er i hvert fald, uanset" and once again relates the argument to the system: " nødt til at have nogle kortfattede navne". And his suggestion is that they stop the discussion and simply accept his suggestion.
S#	Ja, så men lige for min egen orden' sk' - når du så skal have en litteratur liste, for eksempel, over de data hvor gå du så hen?	S# will not immediately accept.The perspective change from a systems perspective to a use and work perspective, about what to do if he wants to create a list of references which is an important product of his work. The mood is interrogative. S# requests information from R# but the modality "lige for min egen orden" is weak. It indicates that S# knows that he is breaking the frames of the demonstration phase making a parenthetical move.
R#	altså, nu viser jeg lige hvordan du peger først, ikke!	R# simply refuses to respond to the request, and makes S# accept the initiated focus on demonstration by asking a tag question, i.e. a question you only can answer positively.
S#	ok, ja.	

R# Der kan du pege på denne her så kommer du over til til til

S# ja

R# til det der - til selve kortet - altså referencekortet - det har jeg altså ikke lavet- det er nemt at lave - så kommer du der over --- og så kan du komme tilbage igen selvfølgelig - og så kan du øh - og notelisterne det er det samme, der kan du også pege, hvis du vil det, der er en note om natur hos Tolkien, så kommer du over til noten der, og så kan du komme tilbage igen.

So now we are back in the demo phase and to the systems perspective: The the system is emphasized and the concepts are from the Hypercard domain ("pege", "kortet", "komme over til"). The mood is declarative.

S# Ja, du kan også komme fra bogen til noten? eller det har du ikke lavet endnu?

R# jammen, nej det har jeg ikke lavet endnu - jeg har kun det her - det er ligesom - jeg forestiller mig at det her

S# det er overstrukturen

R# det er ligesom overstrukturen på det hele.

49

S# starts putting questions again. Trying to understand the system. But since his question goes on something that R# has not implemented yet they both avoid to discuss it.

R#	Ja, emne listen herovre - der kan du ikke pege, der gir' det ikke mening at pege. Hvad sker der hvis man peger? Så gør den bare ingenting. Fordi, det gav ingen mening, ja det kan muligvis	At this instance a conflict arise due to differences in perspective. From systems perspective, R# says that "her giver det ikke mening at pege" i.e. to make sense in the meaning that when you click, nothing happens. Where as from a work perspective, S# says: "yes, it could make sense", i.e. to make sense in the work situation. Due to the work perspective, S# has the initiative for the moment.
S#	Jo det vil det nok jo kunne, fordi du kunne ha' flere referencer, flere bøger eller flere noter til et bestemt emne, hvis du arbejder over en periode, ikke også, du har lagt noget ind med et bestemt søgeord og kommer tilbage til det tema i en eller anden sammenhæng og lægger noget nyt ind i en note.	
R#	Prøv lige at sige det igen!	
S#	altså du kan godt have brug for, at kunne samle hvad der er på et bestemt emne som ligger i forskellige noter.	
R#	ja!, men det gør du ved hjælp af søgefaciliteten	R# can see what point S# is trying to make and can assure him that the problem is taken care of by an other facility then the one he is talking about for the moment.
S#	nå, ja!	
57		
R#	Det kommer. Det her med at pege	But he promise that he will demonstrate it later: "det kommer" thereby reestablishing focus and his own initiative R#
S#	nå, ja!	

Det er simpelthen bare, det er simpelthen bare, du forestiller dig at det hér er tegnet og så den virkelige eller - ref - bogkortet det er referencen, så det som du gør det er at du komme fra tegnet til referencen og det er det du gør hele vejen

S# Jaa

R#og der er jo ingen refencer til emnerne.

R# gets aware of the fact that the conflict is due to the use of two different codes. And he solves the conflict by explicitly telling S# what code he is using. The code of semiotic theory. R# has used semiotic theory in his design of the system. S# also has some background in semiotics so here they can share the code. They use concepts like "signs" and their "reference".

61

R# men der kunne man forestille sig at altså at hvis lavede - hvis man byggede det videre sådan at emner fik en intern hierarkisk struktur

S# Ja, det kunne man forestille sig

R# så kunne man bruge det der, og arbejde med det der. ----

The conversation shortly shifts into an idea generating phase. The modality is weak "man kunne forestille sig". S# briefly responds to the ideas.

R#	Godt! det var pege. Og så har jeg lavet ved dem alle sådan, at man kan skrive, det er jeg ikke sikker på at man skal kunne gøre her, det skal man muligvis gøre andre steder	But the frame "Godt, det var pege", firmly brings the conversation back to the demo phase, with a new focus "Og så har jeg lavet det sådan at man kan skrive ". And in the following there are some quick shifts between demo: "jeg har lavet dem alle sådan", with strong modality and idea generation: "det skal man muligvis kunne gøre andre steder" with weak modality. It looks like the shift in modality is used as markers of politeness. Like R# wants to show that he has thought about alternatives so that he without being unpolity can keep the topic of conversation within the demo phase, with a promise that the discussing is coming later on.
S#	Næh?!	
R#	men nu er der ihvertfald skiftet modus her ikke, og så kan du skrive, hvis du vil det, og du kan kopiere, også hvis du vil det, og det kan du også skrive her ovre.	The frame "men i hvert fald" brings us back to the demo phase. And again concepts belonging to the systems perspective are used: "skrive" and "kopiere".
S#	Ja	R# initiates a small idea generating phase, and S# gets involved
R#	der kan du gøre - jeg ved ikke -	
R#	Det er meget godt at det er her, men det er svært at vide hvor meget du vil gøre det herfra.	
S#	Ja, jeg tror heller ikke at man vil gøre det herfra, -	

R#: Godt! så det var (host) - man altså - ihverfald så skal det være på nogle kort, så skal det være sådan at man faktisk kan skrive heri. For eksempel ikke, hvis du har brug for et ny emne, klassificere en bog, skal du skrive et eller andet ind her. Men det skal nok ikke gøres herfra. - Det er vel også - så - ja - ok-. Så kan man søge --. Nu vil jeg have nogle bøger ud.

but when S# enters the discussion, R# uses the frame "Godt" to indicate that he does not want the discussion anyway. But he continues the discussion with himself until he chooses to go back to the demo phase again. Notice also the HyperCard concepts use:"cards". For the first time R# shifts to a work perspective: "hvis du har brug for et ny emne, klassificere en bog, skal du skrive et eller andet ind her".

In scene 1 the designer, R#, has the initiative. He define the focus of the conversation and his systems perspective is reflected in the emphasis of the systems itself as well as in usage of the concepts of the professional language of the designer. The code has a closer resemblance with the language of the designer than with that of the user. Although the systems perspective dominates, a conflicting perspectives exist, the work perspective of the user. But the designer keeps the initiative by defining and introducing new topics. Frames are intensively used by the designer to control the initiative, particularly with respect to the topic. The declarative mood, and the strong modality, does not open up for the active participation of the user.

Notes on scene 2

In this part of the conversation we have entered the discussion-construction phase where the two systems should be brought together. Here S# shall explain why his system looks the way it does, and give arguments for it to be that way, in order for R# to put the systems together in a satisfactory way. The following conversation shows how R# keeps the initiative and the control over the design process while dealing with S#' suggestions.

R#: Vil du ikke ha flere bibliografiske oplysninger?
S#: jo SB skal med derop, det skal vel osse den der automatisk numrering?
R#: den der, det er den jag ikke vil bruge. Hvorfor vil du ha den egentlig?
S#: nej det er osse...
R#: men hvorfor vil du ha den, hvad vil du bruge den til?
S#: det kan.. det kan jag ikke huske

S# suggests that one function of his system should be kept in the new system. R# does not like the suggestion and just declares so, without giving any arguments for it. Instead he make a request from S# to explain why he wants it. But S# withdraws. R# really wants an explanation because there might be a good argument and pushes again, but S# can't come up with an answer.

R#: den der det var titel og år
S#: sted og år
R#: er det forlag osse?
S#: det tar jag ikke med, det er det nogen det gør men det gør jag ikke
R#: vil det ikke være rimlig nok at ha det med
S#: jo men så ska det i et andet fælt for du ska ikke ha det med i litteraturlisten, der har du aldrig forlag.
R#: men det er nogen tidskrifter der forlanger forlag
S#: mm
R#: så kan man jo godt gøre det at man skrev sted og år i et fælt å så hade et fælt med...
S#: mm
R#: det er tit problem for meg med forlag. Hvis man skriver til et eller andet tidskrift så ska man ta å rote ting å sager igennem.

The perspective has now shifted towards the work process. S# shows firmly that there is a function he does not want, referring to his own work process: "det gør jeg ikke". R# who wants it , almost plagues S#: "men ville det ikke være rimelig". S# gives after with certain conditions, again referring to his working process: "for du ska ikke ha det med i litteraturlisten". R# accepts the conditions and gives an argument for why he wants the function. This argument builds upon R#'s own working process: "det er titt problem for mig med forlag"

In scene 2 the designer initially has the initiative. He asks questions (interrogative mood) to the user, S# who is supposed to argue for *his* demands to the system. When the perspective shifts towards a work perspective, S# for a while gets the initiative. The modality of his sentences is strong and R# can not take the initiative back just with the use of frames, he has to use sincere work oriented arguments to keep the over all initiative.

Notes on scene 3

This scene is from the last part of the session. Now it is no longer a question of what functions should be in the system but how to implement them.

R#: men problemet er jo at hvis du har brugt notens navn til at indexere masser med så skal du rende igennem det hele for å udskifte, kan du ikke se det?
S#: nej men fordi.. altså når du..
R#: men jeg er ikke sikker på jeg forstod det rigtig, altså det du forestiller dig er det at.. Vi siger at vi vil lave en ny note
S#: ja
R#: er det der
S#: ja
R#: å hvad gør man så, så blir man ført over
S#: så laver vi den her "ask" som ligger under noter "Vad er notens navn" å det skriver man ind der, og så går den videre til eventuelt med et spørgsmål mer "Skal du lave en note nu"
R#: "Vil du skrive i den nu"
S#: "Vil du skrive i den nu"

S# wish to implement a function in a certain way to fit his work process, but the focus is on the system. R# gives a system argument. "så skal du rende igennem det hele for og udskifte". S# will not accept it. Now instead of passing the initiative to S#, R# keeps it by inserting questions. (interrogative mood) "Vad gør man så.." that S# has to respond to. S# has entered the systems perspective and comes up with a solution to the problem in terms of HyperCard concepts: "Så laver vi den her ask.. som ligger under noten" og så eventuellt et nyt spørgsmål. Skal du lave en note nu" S# use the word "make" which implies that he falls back into his work perspective. When you work with paper a note is not a note until you have written it so to him to make one belong to the same concept as to write one. In the system though you create empty notes so from that perspective to "make" and to "write in" belongs to different concepts. Here R# actually change S# language usage. And S# accepts it.

S#: ja det gør man så, å så går man tilbage og arbejder videre, så tre dage senere så finder man ud af at den noten sku måske hede noget andet, så går man i notelisten og ændrer det navn, og så vil den her liste også blive redigeret automatisk fordi den henter sit navn deroverfra

R#: nej fordi den heter ikke nogen, den ligger ju fast den her liste

S#: men de.. hvor er inputet til den listen?

R#: det er det du sitter og laver nu

S#: ja men det er ju bare fordi du sænder den direkte op i den liste fra spørgeskemaet i steden for å sende den over omkring notekortet, det kunne du jo godt gøre

S# insists on his solution and points out to R# that it is possible even from a pure systems perspective.

The discussion continues after this pattern until R# tries the solution out and the system brakes down. They both start smiling and the conversational patterns change:

R#: det er simpelthen
S#: du har ikke redigeret..
R#: nej
S#: du har jo fået nye navne her nede ikke osse. De her felter hernede det heter noget andet end de heter i de gamle knapper.
R#: ja de der to de ska fanneme, det er to forskellige mode...
S#: alltså med "gammel" ska du kunne vælge en som skal kunne stå hernede, men det du mangler lige nu det er å lave adressen til de navne her
R#: den går faktisk... den hopper faktisk.. nu hopper den sgu over til noten
S#: men nu er du ude på hovedkortet
R#: det er simpelthen fordi jeg har ændret for meget, jeg vet simpelthen ikke hvad de knapper de gør. Den her "gammel" her..
S#: den betjener, ligesom den over der, den betjener det felt som du har slettet ikke osse!
R#: ja men det gør den sgu ikke riktig. Ska se hvad som sker når jag redigerer den her også så.
S#: fordi de her de heter noget andet nu.

R# and S# now both develop a "hacker" perspective. The emphasis is on what happens on the screen and HyperCard concepts are used. Now suddenly S# takes the initiative. The mood is descriptive. And the modality strong. He stands back giving a recollection of R#s earlier actions. In this situation R# accepts S# initiative there are no conflicting perspectives and no threats to his system. Just a help
to make it run again.

In scene 3 the the focus is on the technical implementation of the desired facilities. The user is more actively involved. There are no open conflicts between the systems and the work perspectives, but in the start of the scene there is a conflict about how to solve the implementation of a facility the user wants. The designer keeps the initiative by asking questions, and when S# slips out of the systems perspective he try to change S#'s code. When the system suddenly brakes down the initiative change. Now user is acting as a logbook, being the one with the overview. There is no discussions of how the system should be. The modality is strong and the mood is declarative. There are no more explicit questions and answers just comments to the state of affaires.

4. Discussion

Even though this particular situation is extremely informal and even though the participants have a lot of shared knowledge beforehand and a great trust in each others competence, there is a conflict between R# and S#; between the one who carries the knowledge of systems development and the one who carries the knowledge of the specific working process. There is a conflict between at least two different codes, the designers system oriented language and the users work oriented language. Here we have seen differences in perspective as reflected in different concepts and emphasis. Furthermore, the designer also has kept the initiative by exploiting conversational rules, for instance frames.

The situation is not an unusual kind of social interaction. You always have to enter a role in a conversation. If it is a situation of great trust you even have the possibility to strengthen the force belonging to your role, which actually makes the conflicts more explicit. For instance you can make ironical meta comments to your own acting like R# actually does now and then: "nu viser jeg lige hvordan man peger først ikke!"

So, is this cooperative design? We certainly believe it is, but then, which kind of social interaction is not cooperative? Only at the point where the rules for conversation are broken the situation becomes non cooperative, for instance when questions get ignored. Our main conclusion on cooperativity is that the important distinction is not cooperative versus non-cooperative but success in cooperation versus failure in cooperation. Furthermore, we believe that the dynamics and creativity of cooperative work is due to differences in background, goals, and perhaps even to competition. Although, the designer somehow should be in charge, he/she is also the one who has the obligation and skill to design systems, but if he/she is not open for new perspectives or is not aware of conflicting perspectives the dynamics might get lost and the user never gets real influence on the system.

So, to support the dynamics of the design situation, the designer could be aware of the different perspectives, make them explicit and exploit them. Moreover, we would like to draw to the attention of the designer the various aspects of the discourse, for instance frames, change in focus, interrogative request from the user.

In practice, it is not easy to change ones language usage. You can't change a conversation the same way you change a Pascal program. Moreover, when you are involved in a conversation it is not easy to reflect on the situation at the same time. An indirect way of changing the pattern of conversation, is to change the way the design situation is set up. For instance, in the specific situation, if the prototype had been simpler it would have been easier for the user to try it out himself, for instance by sitting at the keyboard himself. For a more elaborate discussion of the design situation, particularly the design tool itself, we refer to [Andersen ét al 1987] and [Bødker ét al. 1988]. For the discussion of a different and instructive case we refer to [Bødker & Grønbæk 1989].

Our final message to designers is to reflect on their own practice and to use the design situation of this paper as a reminder or paradigm case.

References

Allwood, J. *Linguistics as Action and Cooperation: A study in Pragmatics*. Department of Linguistics University of Gothenburg 1976.

Andersen, P.B. ét al.: *Computer Support in Cooperative Design and Communication*. DAIMI IR-70, Aarhus Universitet 1987.

Berry, M. Is the teacher an unanalyzed concept? In Halliday & Fawsett: *New developments in systemic linguistics.* Frances Pinter. 1987.

Bødker, S. and Grønbæk, K.: *Cooperative Prototyping*. Department of Computer Science 1989.

Bødker, S. ét al.: Computer Support in Cooperative Design, indeholdt i *Proceedings of the Conference on Computer-Supported Cooperative Work*, Portland 1988. Også som DAIMI PB-262, Aarhus Universitet 1988.

Coulthard, M. *An Introduction to Discourse Analysis.* Longman 1977.

Einarsson, Jan. Blandskolans femma. Iakttagelser under en skoldag. *Språk och kön i skolan 5.* Lunds universitet. Institutionen för æmnesmetodik och æmnesteori, avdelningen för svenska 1981.

Halliday, M.A.K. *Language as social semiotic.* Edward Arnold 1978.

Holmqvist, B & P. Bøgh Andersen. Work Language and Information Technology. In *Journal of Pragmatics 11* 1986

Johnson, B. & Weawer, G.: Using a computer-Based Tool to Support Collaboration, in *Proceeding of the First Conference on Computer Supported Cooperative Work*, Austin Texas 1986

Sørgaard, Pål. A cooperative work perspective on use and development of computer artifacts. In P. Järvinen(ed.): *The report of the 10th Information Research Seminar in Scandinavia*. Tampere 1987.

CREATIVITY AND SYSTEMS DESIGN

Kristo Ivanov

Umeå University, Institute of Information Processing

S-901 87 UMEÅ (Sweden) – Tel +46 90 166030 – Fax +46 90 166688

Abstract: One main purpose of this paper is to foster among all of us, by means of a non eclectic attempt of synthesis, a consciousness about the limitations of our own approach and interest for others' approaches as a way to better design, that is, creative design.The paper is molded in the form of a "reader" and a guide for studies through a great amount of potentially relevant literature with emphasis on the Scandinavian scene, but with no claims of full coverage. It indicates in which way such studies relate to design and creativity, and how they can contribute to improve systems design. After a short introduction in the way of summary (chapter 1) it starts by dwelling on the meaning of creativity (chapter 2), not for defining it with the purpose of applying a deductive schema on it, but rather for amplifying it and for suggesting the broader context in which it should be examined. It will later (chapter 3) touch upon some work which has been done on the issue in terms of four key concepts: design, change and dynamic modeling, cooperative tools, and critical social theory. This is followed by a survey of both opportunities (chapter 4), and dangers (chapter 5), of emphasis on creativity in information systems research. Finally it will present some conclusions (chapter 6), and suggestions for future research (chapter 7). One conclusion is that the growing emphasis on creativity, flexibility and change in an eclectic spirit may be a reaction against, and a symptom of, superficial approaches to insufficiently understood problems that are conditioned by the disregard of aesthetical and ethical aspects of computer and information science.

Keywords: Creativity, design, cooperative work, distributed artificial intelligence, flexibility, change, evolution, dynamic modeling, decision support, knowledge augmentation.

Before the modern era all ethics used to begin with "Thou shalt not." During the industrial age, the ethical code shifted to the idea of "Be productive." The ethics of the computer time world is "Be creative." Evil in the information age is viewed as anything that places obstacles in the way of the innovative act, the novel experiment, the untried scenario.

J. Rifkin

Possibility [in word processing] dominates over consistency of vision - at least as consistency was envisioned by the book tradition and by philosophical mentality. But this is not possibility in any grounded sense of taking up a certain kind of existence based on a distinct context or felt necessity. On the contrary, such a proliferation of possibilities remains free-floating because they remain under the supervisory control of technical power. The possibility field is akin to the idle curiosity that can paralyze any effort to effectively come to grips with things. With word processing a creative super-abundance prevails over the composure of the mind characteristic of traditional formulation. Mental excitement and stimulation supplant mental composure.

M. Heim

1. Introduction

Flexibility, evolution, development, change, design, knowledge, intelligence and particularly artificial intelligence are today almost as popular buzzwords as systems and information. When bringing in creativity into computer and information science it is therefore appropriate to start by dwelling a while on the meaning of the word, not for defining it with the purpose of applying a deductive schema, but rather for amplifying it and for suggesting the broader context in which it should be examined.

2. Definitions as Amplification

Following the cultural patterns represented by dictionaries, "to create" means to bring into existence, to make out of nothing and for the first time; to cause to be or to produce by fiat or mental, moral or legal action; to produce or effect as an act of grace; to bring about by a course of action or behavior; to cause or occasion; to produce; to design; to make or bring into existence something new; invent. To create is then to cause to exist, to produce, to bring into existence, to be the occasion of, to originate, or even to invest with a new character, office, dignity or position. Creation is a production of art, craft or intellect. The close word "creationism" suggests some other connotations of the phenomenon that we try to grasp by reminding us of the doctrine that a human soul is created for each human being at birth; it is the theory that the universe was brought into existence out of nothing by God, and that new forms and species are the results of special creations. It is then clear that creativity is related to the debate, revived some years ago, about creationism versus evolutionism. Does it make any difference whether we attempt to create new information systems or let them evolve through subsequent changes?

We saw above that a close synonym is "to invent", which means to search out of come upon, find, discover; to think up of imagine: concoct mentally, fabricate; to create or produce for the first time, be the author of, devise, originate, contrive. Invention is then an act of finding or of finding out; a faculty for creative selection of theme and imaginative treatment of design or content; an act of mental creation or organization, application of knowledge; a product of thought or mental synthesis, an idea or concept; the creation of something not previously in existence. To the extent that the creativity is related to "new" we may also think about the implication of being different or distinguished from, to be original.

In the wealth of the above connotations it may be difficult to establish some particular emphasis on the difference between e.g. invention and discovery, between creation and acquisition of knowledge, or between productivity of the knowledge process and the value of the product (Lerda, 1988). Such attempts as well as distinctions between descriptive and prescriptive models are probably akin to the classical scientific debate about the IS versus the OUGHT and are then an expression of positivistic tendencies as denounced by several writers (Brecht, 1941; Churchman, 1971, p. 201f; Cranberg, 1968; Forsgren, 1988). In spite of the fact that "produce" is mentioned in the explanation of create I will keep a distinction between creativity and productivity by emphasizing the mental, imaginative, moral and legal dimensions in the definitions. There is even a religious dimension in that one may speak about "to produce or effect as an act of grace", something which is clearly understandable in terms of e.g. analytical depth psychology where intuition and the unconscious are related to divine archetypes.

3. Some Previous Work on Creativity

Earlier work about creativity in information systems should properly be historically and scientifically related to research on creativity in general. Creativity has certainly been also an educational pedagogic matter and as such it has certainly followed the history of education, being in turn cathechesis, philosophy, psychology, and sociology.

Creativity (as seen in an encyclopedia of philosophy) is sometimes understood as the individual capacity, potentially present in the most varied fields (from artistic or scientific production to everyday life), which consists in grasping the relation between things or ideas in a new way, or in formulating intuitions which were unforeseen in habitual or traditional thinking schemes. In psychology two main schools are mentioned: a cognitive (J.B. Guilford) and a clinical-psychoanalytical (E. Kris, cf. his *Psychoanalytic Explorations in Art,* 1952). In Sweden "percept-genetic analysis" has attempted an integration of psychoanalytical and experimental approaches which, in the context of creativity, resulted in a definition of creativity as a tendency to break conventional perceptual patterns (Smith, 1986, the last chapter, and The Creative Process, in press; Smith, 1987). Creativity in industrial settings has been studied under the labels of renewal and innovation (Ekvall, 1988). Creative thought has also been named "open", and it coincides with the "divergent" thought that Guilford opposes to "convergent" thought. There are many psychological tests of creativeness with results that generally do not correlate with the results obtained from tests which measure the ability to solve strictly logical problems. In contemporary non positivistic epistemology (e.g. in M. Polanyi's book *The Tacit Dimension,* 1967, as well as in social systems theory), in opposition to the case of inductivistic-positivistic epistemology, knowledge of the psychology of the creative process is considered to be fundamental for an adequate elaboration of a theory of scientific discovery (Chandrasekhar, 1987; Koestler, 1964, pp.255ff.). What is true for this non positivistic epistemologys seems also to be true for certain currents in research on styles of problem solving and in modern systems science close to the interface towards information science (Ackoff, & Vergara, 1981; Hill, 1989; Kerola, & Taggart, 1982; Strzalecki, 1989). A wealth of insights is thus adduced that is not possible to integrate in this paper but should be considered in our future research (Amabile, 1983; Barron, & Taylor, 1975; Ghiselin, 1952; Gilchrist, 1972; Gough, & Woodworth, 1960; Harman, 1984; Pearl, 1984; Peat, 1984; Scandura, 1980).

In the specific field of information systems research no explicit development of the issue of creativity is known by us with the exception of a handbook-type rule of thumb summarizing reference to general research about the matter (Goldkuhl, & Röstlinger, 1988, pp. 46ff.) and to popular classics on creativity (de Bono, 1970; le Boeuf, 1980). Creativity was discussed in its relation to computers was at a conference on Culture, Language and Artificial Intelligence (held in Stockholm in maj-june 1988). The discussion (Östberg, Whitaker, & Amick III, 1988, see pp. 17-20 for comments) was, however, directed at AI. It was also kept at a rather general and anedoctic level on the basis of some empirical material, without great theoretical ambitions. Such rule of thumb presentations usually introduce different steps or phases of the creative process, often with the implication that they represent some kind of method which can be followed in order to enhance creativity. Examples of such phases are 1) Preparation, 2) Incubation, 3) Illumination, 4) Verification, or 1) The initial ideational phase, 2) The intersubjective, 3) The analytic, 4) The synthetic, 5) The communicative. The latter division of the creative process in 5 phases, however, was originally based on a deeper and more comprehensive research (Buttimer, 1983). It contains a very broad and ambitious general overview of literature relevant for creativity, as it also can be found in some studies of the psychology and creativity of science (Mahoney, 1979). That part of the material which most explicitly deals with communication and information, however, (Buttimer, 1983, pp. 95ff) is clearly insufficient for illustrating the creativity problem in the context of information systems research.

There is also a book that certainly displays theoretical-educational ambitions in relation to computer supported creativity of children's education (Papert, 1980). Unhappily there is no space here to introduce the serious criticism that it deserves, since it would also require a special development to show its relevance for reasoning about creativity in information systems development and research.To the extent that creativity is expected to be more clearly related to design and change of information systems, however, there are several approaches that will be considered below.

3.1. Design

The word design in our culture has an important engineering connotation, and its present use may sometimes express an engineering-utilitarian ideal where creativity is often seen as a means for productivity and profit. It will in fact be noted that the concept of design as, for instance, in computer systems design or information systems design was characterized during the first decades of computer and information science by a typical engineering "life-cycle" model, roughly: 1) Design or rather design specification, drawing, construction documents; 2) Physical construction, manufacturing, implementation, installation; 3) Operation, test or evaluation, control decisions on scrap/rework, followed by maintenance or iterative design.

The criticism of this life-cycle model has been thoroughly summarized during the last decade (Ehn, 1988; Forsgren, 1988; Sørgaard, 1988) but the scientific and philosophical basis of the model, for a subsequent judgement of the alternatives, has seldom been made explicit, besides some more or less superficial references to Taylorism, positivism or rationalistic systems theory. The life cycle model at its best may indeed be considered as a special case of the "interactive" hypothetical-deductive experimental method in natural science where the design specification corresponds to the hypothesis, physical construction or manufacturing corresponds to the experiment itself, and control corresponds to the evaluation, conclusions and actions derived from the experiment. The obstacles to creativity in this context are represented by the controlling-manipulative ambitions of the experimental method in natural science. It attempts to impose a tight isolated one-way binding between induction and deduction, a methodological "murderous" straitjacket that sometimes appears under the guise of "the big project", and leaves untouched the creative question about the origins and value of the hypotheses and their relation to the freedom of the ethical subject (Stolterman, 1989) who then needs a "guide for the perplexed" (Schumacher, 1987).

The development of this type of thinking in order to relate it to social science (Churchman, 1948) led to the conceptualization of experimental design turning into design of inquiring systems (Churchman, 1971) and into planning (Churchman, 1979). Such an expansion can take, for instance, care of the fact that design may be preceded by market research with due consideration of feelings and conflicting social values, and implementation may consist of a juridically binding sale-purchase trade-off contract based on specifications and tolerances, etc., akin to some aspects of conversational negotiation models and transaction costs theory (Ivanov, 1972, pp. 4.36, 4.44f; Ivanov, 1986, p. 46-52; Ivanov, 1987).

The expansion of life-cycle models of design into planning and inquiring systems has the merit of evidencing further that the older traditional life-cycle, understood as a particular case of hypothetical-deductive or inductive-deductive schema, was in turn a first attempt to synthetize British empiricism and induction, as represented by John Locke and others, with continental rationalism and deduction, as represented by Leibniz, Descartes and Spinoza. This insight was obtained by using the idea of design to reformulate some classical philosophical ideas, since classical philosophy defined epistemology to be the idea of speculating on how the human mind comes to acquire knowledge. The revision of this formulation was to ask whether it was possible to design the process by which knowledge can be acquired, using both humans and nonhumans (e.g. computers) in the process. Such process would correspond to the aims of present so called artificial intelligence, AI and expert systems, which were earlier labeled as "intelligent technicians" (Churchman, 1971, chap.1, 4).

The understanding of the design concept as originating in this manner from engineering in order to be expanded into human sciences and philosophy has the advantage of immediately identifying the issue of advanced design as being the well studied issue of synthesis between rationalism and empiricism, which in the history of computer science area are materially represented by so called management information systems and data bases (Ivanov, 1988). In this way it will also be understood which are the outlooks of a design that is based on American pragmatism, experimental idealism and dialectical social systems science (Peirce, Singer Jr., Churchman) and dialectical idealism or dialectical materialism (Hegel and Marx) that programmatically attempted such synthesis. Immature synthetical attempts are represented in the computer field by the rush for AI and for "general purpose expert systems" (sic!) and related dubious trends (Nagel, 1988).

It would then be noted that the rush for more sophisticated socially and politically oriented design methods such as user centered cooperative design of computer tools for improving workers' skill and for a better work environment, involvement of trade unions, phenomenologically oriented design, etc. fits into the structure of the continental cultural debate during the last several hundred years. What we may be witnessing is then a computer oriented design version of the clash between enlightenment and romanticism, between liberalism and socialism, a clash that tends to become apparent only in extreme matters of control of workers, privacy and data bases for central state planning (Ivanov, 1986, p. 50) The introduction of such new concepts such as tacit knowledge or personal knowledge, for example, will mainly recall on the scene the controversies of the continental romantic movement but without benefitting of the insights into the issue which are offered by modern pragmatists (Churchman, 1971, pp. 10-11, 18, 21, 28, 82-4, 87, 88, 92, 118, 137-8, 154-5, 158, 162, 171-3, 175, 177-8). The prospects of finding a better design process are then not better than the prospect of solving such clashes and bridging the gulf between the natural and formal science of the high-tech computers, and the human science for their use.When issues attain such a level of complexity it may be legitimate to wonder whether the road to better creative design is through more sophisticated understanding in terms of richer theories and "a priori" representations (Churchman, ibid., pp. 133-146), or whether the right and realistic road should follow conversational constructive common sense (Churchman, ibid., pp.104-5, 115, 119; cf. Forsgren, 1988) , or plain ordinary language. In the latter case the ultimate issue would, of course, be how to foster and ascertain what is to be considered a "good conversation", especially compared to a good revolution.

In summary, a pioneering conception that appeared well before creativity, design, cooperative work, artificial intelligence, etc. had become fashionable is "the design of inquiring systems" (Churchman, 1971, where the concept of creativity itself is considered on pp. 3f, 17f, 143, 205). The word inquiry is used instead of information with the explicit purpose to emphasize epistemological, cooperative and political aspects of the question (cf. ibid,.p.159ff). In that context the avowed purpose is to study the creativity of science, i.e. actions that lead to new knowledge. It is possible to recognize some of the above understanding of creativity as hypermedia-like "grasping the relations between things or ideas in a new way" as corresponding to what otherwise is labeled as "Leibnizian inquiring systems" (ibid., pp. 19ff). The research into inquiring systems places design and creativity together in order to examine their similarities and differences. It is a search for the method of identifying the creative act and an attempt to design creativity in various processes which include the presently fashionable knowledge topic of knowledge representation (ibid., p.143).

The pragmatically influenced approach to design runs into considerations about the image of the hero in archetypal analytical psychology (Churchman, 1971, pp. 202ff). It is, however, possible that such a theory of design must be developed further in order to cope with some of the intrinsic limitations of the pragmatic heritage as suggested in different ways (Ehn, 1988, pp. 184-232; Heim, 1987, pp. 228ff; Ivanov, 1986). Some work in this direction suggests that we may still be able to talk about a creative and integrative theory of design, while at the same time questioning the present trends towards increased emphasis on method at the expense of ethics and aesthetics. We also expect that a new proposed programme for interactive design (Nilsson, 1987; Nilsson, 1989) will ultimately be able to take into account the lurking dangers of man-computer interaction whenever the computer is considered as an artifact or tool, instead of as an instrument, originally and insightfully described, for the context of word processing, by Heim (1987, pp. 198-213, 233-246).

One particular approach in design has been the so called human-scale information systems (Nurminen, 1988). It was written before the present new emphasis on creativity. Creativity may therefore be seen as implicit in the concept of personal autonomy represented by such visions as "Each worker has his or her own personal system, whose limits are defined in terms of that person's job", so that the workers can themselves make decisions as to changes in their systems, fostering their ability to cope with various situations and developing the capacity of those concerned to solve similar problems by themselves in the future, instead of offering them a ready made solution (ibid. p. 116-117, 126, 129, 149). This conception may be seen to roughly correspond to "Singerian inquiring systems" where the ultimate ideal is the "unified decision maker, client, and designer" (Churchman, 1971, p. 201-204). When it comes to integrate the efforts of the autonomous units, as it is necessary in order to differentiate a chaotic conglomerate from any organization, the attention is shifted towards the building of bridges and dialogues between different paradigms of formal-natural-technical science and human science (Nurminen, 1988, pp. 11, 51, 173, 175, 178, 180, 186-7, 192). The whole effort, however, risks to get reduced to an oversimplified eclecticism represented by the single person who "has competent knowledge of and is able to make use of various alternative 'objective' theories", knowing that "in any practical situation the solution to a particular problem should be dictated by a particular theoretical paradigm", and that one should "adopt different perspectives and choose the one most appropriate in a given situation" with the "possibility of choosing or changing one's paradigms" (ibid, p. 178, 180, 187). In order to avoid running into ethical relativism and situational ethics this approach would probably benefit and gain credibiblity by being related to the very close aims of the continental romantic and postromantic movements in the past century (Bortoft, 1986), and by incorporating the historical critique of eclecticism suggested by Ivanov (1988, p. 97). What can be rescued out of the interest for eclecticism will at any rate be, of course, that which enhances "apperception", i.e. the designer's ability to design the system from many points of view (Churchman, 1971, p. 75).

3.2. Change and Dynamic Modeling

Following the general orientation of the design of inquiring systems an early attempt to address cooperative creativity in terms of novelty and change was made in terms of the metaphor of "quality" or accuracy-precision as related to negotiation (Ivanov, 1972) that was later summarized in other work (Ivanov, 1986, pp. 47ff; Ivanov, 1987). Such a conception relates to the understanding of creativity (see definitions above) as having a focus on novelty, where "new" is seen in the sense of "different or distinguished from". This particular approach has some integrative bridging characteristics in the sense that it connects the soft issue of creativity and change to the hard realities of measurement in natural science and in industrial manufacturing. In its emphasis on contradiction or breakdown, negotiation, agreement, and contractual law it also has points of contact with other British (Atkinson, & Checkland, 1988) and Scandinavian (Mathiassen, & Nielsen, 1988; Mathiassen, Rolskov, & Vedel, 1983) approaches to design, including the interest for transaction cost theory.

The idea of quality to be attained by means of social differentiation and communicative pluralism in measurement and in data processing in the spirit of Hegelian-Singerian inquiring systems (Churchman, 1971; Ehn, 1988, p.187) was later reintroduced through the metaphor of cooperative creativity or "constructivity" (in Swedish: samskapande) in the direction of cooperative design for flexibility and change, with emphasis on dialogue or conversation pioneered by Nissen (1976) and later taken up by others (Forsgren, 1988; Levén, & Nordström, 1989; Lyytinen, 1986). Here the criticism of the concept of ideal speech is relevant (Ottmann, 1982). Creative cooperative change or flexibility would be enhanced by means of five strategies: 1) Metaphorical, 2) Heuristic, 3) Documentation, 4) Computer application, and 5) Education (Forsgren, 1988, pp. 125-162, 176f). As in other attempts at creative design that have been made in other academic traditions of information research, such as "change analysis" and "continual-constructive systems development" (Forsgren, 1989; Goldkuhl, et al., 1988; Nissen, 1988), there is a general trend in attempting to develop the issue of creative design in the direction of approaches that incorporate more technicalities, in the sense that they are more specifically dedicated to computer applications. Such technicalities, is hoped, will work as a "motorcycle" in the sense of "Zen and the art of motorcycle maintenance" (Pirsig, 1974, where creativity is also addressed as quality). Non philosophically inclined people are supposed to get lured into the deeper issues of creative design at the interface between the classic and romantic perspectives.

In most contexts of research on use of computer technology, however, creativity is treated only vaguely, e.g. by remarking that it consists of an integration or swinging between analytic/logical and intuitive thinking: "The integration is a prerequisite for obtaining high quality solutions" (Goldkuhl, et al., 1988, p.48, quotation).

A rather original approach to the problem of change in terms of flexibility in the user's learning of technicalities has been presented under the label of "Levels for Utilization of Computing"

(Järvinen, 1985). It conceptualizes a computer systems design which allows for six levels of computer utilization in increased degree of technical sophistication: 1) A user knows potentialities and restrictions of computer, 2) A user can present report requests, 3) A user can initiate programs, 4) A user can regulate programs, 5) A user can program, 6) A user can develop new systems software including compilers, utilities, macros, etc. The ladder of learning from 1 to 6 strives for providing to the user increasing freedom in computer usage, responsibility for decisions and for correctness of programs, freedom from work instructions - control of work pace, and enlargement of human capabilities in terms of processing capability or reasoning power. One can read into this scheme an implicit understanding of conditions leading to increased creativity as well as a systematics for an evaluation of the present CASE wave of fashion. One questionable assumption seems to be that such creativity is a rather linear function of only formal-technical knowledge. A later conceptualization bases creativity, in terms of autonomy, on "mathematical systems dynamics" (Järvinen, 1988) but its formalism unfortunately still keeps it a safe distance from social science and from its ongoing debates (Ivanov, 1988, p. 99, comments on the position of the originator of mathematical systems dynamics).

3.3. Cooperative Tools

Besides the above mentioned trend towards cooperative creativity there is in Scandinavia, in keeping pace with the political and technological development, a trend from general strategic and organizational design towards creative tools that has been overviews in a recent work (Ehn, 1988). "Work-oriented design of computer artifacts" (ibid.) will certainly claim to have addressed the issue of creativity. It is said to suggest a design approach but it revolves around the concept of democracy and skill, the meaning of the latter being definitively related to *given* tasks, recalling the discussions about intelligence vs. personality vs achievement tests in psychology. The more explicit connections to creativity are symptomatically relegated to an epilogue on postmodern reflections (ibid., pp.471ff). Eclecticism is admitted with certain embarassment, as it is admitted that conflicts between approaches inspired ex-post by Heidegger, Marx, and Wittgenstein have not really been resolved. The degree of integration between different philosophical and ideological trends is indeed negligible in comparison with the one reached by pragmatism in what concerns e.g. empiricism and rationalism. Furthermore there are important recognized connections between marxism and pragmatism, while it appears more farfetched to search for such connections in Wittgenstein and Heidegger. And still, such an integration or the connections are indispensable for making some sense out of the agglomerate of texts and practical experiences on physical manufacturing, mainly with trade unions in the graphic industry, which constitute the pride of this approach.

There is, nevertheless a conclusion of work-oriented design of computer artifacts (ibid.) and it is that we should "revisit" Bauhaus functionality and postmodern playfulness with reference to "a semiotic play with signs", but we should do so in a spirit of "lost innocence", i. e. a consciousness of the failure of Enlightenment's rationality, liberalism, and communism. Yet, in spite of lost innocence, we should paradoxically keep faithful to the emancipatory ideal inherited from the Enlightenment and represented today by the trade unions, a belief in progress, work, and democratic rationality [cf. the Singerian trilogy of "production-science-cooperation" and "the heroic mood" in Churchman., 1971, p.201-204].

In contrast to work-oriented design of computer artifacts, computer supported cooperative work (CSCW) has more limited and more technical ambitions, in spite of risking to become a new buzzword (Bannon, & Schmidt, 1989), like distributed artificial intelligence (Bond, & Gasser, 1988). Its concreteness relies, for instance, on object-oriented programming and computerised shared material (Søgaard, 1988). It is expected that computerised shared material will enforce some kind of anonymous structural coordination to the extent that this material simulates physical material, and as such compels users to take cognizance of the consequences of each other actions when working simultaneously on it. The approach apparently relies on the idea of finding a pivotal object or idea, in figurative terms a fulcrum for the lever, a point of balance between morphological structural and functional classes (Churchman, 1971, pp. 43-46). As such it assumes a kind of Newtonian mechanics and mechanics of non-deformable bodies or, more generally, an analogue to the "invariants". While the idea may be useful for coordinating Newtonian mechanical applications like inventory control and scheduling of physical resources, it leaves untouched the question of "Singerian" cooperative creativity in an organizational setting (Churchman, 1971). It certainly requires more than a Newtonian framework for design. It would require finding meaningful formal invariants within the frame of advanced social theory. The hope expressed by the computer supported cooperative work approach is nevertheless to implement an empirical setting that preserves imagination and intuition (Søgaard, 1988, chap.7, p.45), and finding a balance between the need for more emphasis on design and the need for standardisation, or finding an unspecified "dialectic between the analytic and dialectic approach" (ibid.with quotations, pp. 49ff.) Standardisation, however, is a rather complex matter (Churchman, 1961; Guillet de Monthoux, 1981).

3.4. Critical Social Theory

Lately there has been an increased interest for critical social theory as related to the field of informations systems research. Its main results up to now appear to be some interesting and ambitious classificatory and comparative studies of different research approaches, leading to suggestions for future socially oriented research. The connection to creativity is not clear or at any rate is not explicit. It may be assumed that in this approach a creative situation, i.e. which fosters creativity, corresponds to the "ideal speech" situation. In spite of the fact that the heavy criticism against a particular critical social theory (of J. Habermas) has been overtly recognized (Lyytinen, 1986, pp. 163-169), surprisingly enough no real consequences have been drawn from it. The crucial problem of achieving the prerequisites conditions for a rational discourse or ideal speech as also aimed at by others (Forsgren, 1988; Levén, et al., 1989) is addressed neither at the social level of political science, organizational policy and legislation (Ivanov, 1986, p.58f), nor at the individual level of ethics and religion . This does not seem to prevent, however, the conclusion that Habermas' critical social theory enables to understand the social complexity of systems development and that "the recognition of the need for an ideal speech must emerge as a binding norm both societally and individually. Only in this way the truth, justice and the good life can emerge" (Lyytinen, 1986, p. 169f.).

I myself prefer the equivalent classical form of the last statements such as "The recognition of the need for God must emerge as a binding norm both societally and individually", as God is "operationalized" in the holy books and in literature that integrates science and politics with theological questions (Blumenberg, 1985; Buckley, 1987; Churchman, 1971; Ferré, 1987; Ivanov, 1986; Jung, 1953-1979; Lewis, 1988; Riley, 1986). Such conception is consistent with one above mentioned dimension in the definition of creativity that no other approach succeeds to lend any serious meaning to "to produce or effect as an act of grace". One main problem of the critical social approach is, of course, the neglect of precognitive and preverbal dimensions of consciousness (Ottmann, 1982), and the sizable gap between the technological-scientific and practical-discoursive rationality, which is much better bridged by the pragmatistically influenced social systems theory. My own research starts from social systems theory (Churchman, 1971) and is oriented towards such precognitive and preverbal aspects, or rather unconscious and intuitive aspects, with the purpose of bridging the above mentioned gap (Ivanov, 1986). It may also turn out to be possible to incorporate the relation between Churchman and Habermas (Ulrich, 1983).

4. Opportunities and Difficulties

As a meta-catchword for catchwords on this occasion we may think about, say, critical design of cooperative creative tools for design of flexibility and change. It is not necessary to dwell at long on opportunities since the whole information society is busy in spelling the issue. As recently expressed without references to Marxian perspectives (Rifkin, 1987, pp.235f), a new generation of worker owned and worker operated companies may integrate the needs of the individual workers with the imperatives of the production process. In democratically run enterprises group participation in the design of the work process may ensure greater emphasis on face-to-face interaction with the help of appropriate small-scale technologies, tools that are designed to work in tandem with the biological rhytms of the human body. Efficiency will have to be substituted by sustainability implying a new kind of productivity. It is, however, not obvious and it has not yet been demonstrated that this can be implemented by some new kind of *computer* tool technology and computer systems as recently suggested (Ehn, 1988, who dwells on the thema of "the toolness of computer artifacts" on pp. 417-444), having the inbuilt presupposition of modern formal science, and having the presuppositions of speeding up, or at least increasing the productivity of the work process .

When it comes to a more concrete level of claims, the claims for creative tools for creative democracy and skill reveal interesting paradoxes. In traditional handicrafts a tool is a tool or a "pure-transparent" means of production from the point of view of the single craftsman who does not need to, but is able to, reflect on that tool while producing an object under normal circumstances. Let us assume that the degree of transparency enhances some kind of creativity even if skill and creativity are not the same thing. Transparency has probably been developed by the guild during a long time of work on an opaque non symbolic world. Tool transparency for the guild member has then contributed to maintain him in an position of relative power in society.

A paradox today is that those who want to develop transparent easily used computer tools or artifacts at the same time contribute to the destruction of fundamental privileges of the worker-user. This worker will discover that in the name of easy democratic knowledge "for all" he has become easily replaceable by anybody else who can maneuver the powerful transparent tool of the future. He will also discover that the power which he and his fellow workers harness by those tools only works purposefully on an opaque world of physical reality, and only under routine circumstances dictated by an elite. The only tool-independent power which will be left is the gradually weakening power of lobbying and of collective strike, bringing the political and economic status of trade unions to the fore. The worker will find out that the *purposeful use* of powerful computer tools requires system-coordination and "domain" knowledge about the field of application, and that this knowledge is only indispensable for, and the prerogative of, an elite of *toolmakers*. In other words, it will become clear that what is a transparent tool for the worker is an opaque instrument for the knower who can afford to learn about instruments rather than about tools. Vague references to personal or tacit knowledge, or to eclectical tool boxes, or to playfulness at work will not do and will not contribute to solve the paradox. Consider also other complications: let's suppose that somebody invents a new musical instrument, say, a violin; what about the criticism that its user interface is too complicated and requires many years of training to learn to play? (M. Nurminen, personal communication, May 9, 1988; also reported to originate from Bill Buxton, Xerox Europark). Lately some proposals have been presented for the study of these complications, suggesting the relevance of the philosophical work of G. Bachelard (Nilsson, 1989). Basic assumptions of Scandinavian interpretations of phenomenology that are found in this context (the "Winograd-Flores tradition") begin also to get challenged (Nilsson, 1989; Whitaker, Östberg, & Essler, 1989). Powerful physical instruments as seen by a physician or by a military commander who activate a nuclear reactor or an atomic weapon may be transparent to these particular individuals but they will certainly not represent democratic knowledge. Their transparency will be transparent only in a particular coordinated or organized operational setting. The debate about computer systems for "star wars" as summarized in various contexts (Borning, 1987) indicates, however, that it is doubtful whether any complex operational settings will ever be attained and be discriminated from creative *experimental* planning settings (Forsgren's "constructive" settings, or Nilsson's "interactive" settings).

The "toolness" of computer artifacts as applied in social contexts will therefore be dependent upon the same considerations that have besieged the historical debate about creative experimentalism vs. planning, and about instrumentalism (Churchman, 1979; Ivanov, 1988). Much of what has been written about the computer as a tool or artifact for creativity could become highly problematic if the word instrument were substituted for the word tool. It is, in final analysis, also a question of context for the tool, wholeness, organization, theory, volition, aesthetics, and ethics. Early pragmatism, as the latest one to which we refer (Churchman, 1979) established definite relations between creative education or education for creativity, aesthetics, and ethics (Heim, 1987, pp. 230, 288, with reference to Beyond Creating, as an alternative to "pragmatic" emphasis on doing and on unreflective creativity). The assumption (Ehn, 1988, p. 57) that practical skill is more fundamental than theory and rationalistic thinking is also an assumption of utilitarianism shared by both British empiricism, liberalism, socialism, and Leninistic communism: "Theory, my friend, is gray but the tree of life is eternally green", or "Practice is one hundred times more important than theory" (Johnson, 1987, p. 86, quoting Lenin. Also attributed to Goethe). It would be interesting to compare the cultural background of these quotations with the background of other apparently equivalent statements in the German postromantic tradition, dealing with the development of theoretical thought within the human waking-consciousness, the conflict between Being (existence, "being there" *Dasein*), and Waking-Being (waking consciousness, *Wachsein*), the antithesis represented by the phrase 'thought and action', where "life *makes use* of thought", but "all knowing of Nature, even the exactest, is based on a religious faith" (Spengler, 1981-1983/1918, vol.2, pp. 7, 11, 500, and vol.1, p. 380). Deeper and probably less controversial insights could be gained in this area from the work of undeservedly but symptomatically ignored thinkers such as F.P. Maine de Biran (1766-1824) with his theorizing about habitude related to manual work and mechanization, F. von Baader (1765-1841) with his theological criticism of Kant, L. Ollé-Laprune (1839-1898) and M. Blondel (1861-1949) with their theorizing about reason in its relations to will and action, close to the main concepts of pragmatism.

5. Dangers

A great deal of modern work on creativity in information systems relies on a linguistic approach as a basis for cooperative communication. It has been suggested that the contemporary "linguistic turn" in philosophy (Rorty, 1967), including the work of Derrida, Habermas and Barthes, is but an intellectual byproduct or a correlate of the contemporary interest in information and communication, and it has been pointed out that a powerful critique of semiotics and its relationship to language theory has been raised by different authors (Churchman, 1971, pp. 105, 123-5, 145, 195; Rifkin, 1987, p. 267). Meditative presence of mind and contemplative capacity are often incorrectly associated with selfishness (but, paradoxically enough, not with the otherwise appreciated "autonomy"), when in fact such presence of mind is the necessary condition for the quality of a true communication. Much extroverted and busy togetherness fits into productivity but ultimately undermines the ability to be together, and threatens to make mechanically repeatable thought, as in word processing, devoid of personal presence (Heim, 1987, p. 240). The privacy of the solitude of writing is disrupted by the possibilities to keep "promiscuously" on the public cooperative network: the importance of being alone in the private act of creativity (Hill, 1989) is easily disregarded. The apparent autonomy of desktop self-publishing disrupts the delicate social fabric which had grown as a backbone for social communicative book and journal interaction, opening opportunities but also new threats to the freedom of expression (Heim, 1987, pp. 215ff). The computer time frame may destroy the traditional temporal skills which allowed people to relate to each other intimately by synchronizing individual and group behavior (Rifkin, 1987, p. 28, 74ff, 223ff), in spite of asynchronization being sometimes equated to "freedom". Some systems for computer aided design CAD or so called interactive decision support systems accelerate the flow of work-related activity demanding from the designers decisions at such a rate that their creativity has been reported to decrease dramatically (ibid., p. 138f). The solution in terms of a change of attitude towards an organismic ecological participatory organization of work and society (as suggested by Rifkin) is, however, mainly a weak analog to certain historical religious attitudes including pantheism (Blumenberg, 1985; Buckley, 1987; Riley, 1986). To the extent that creativity is equated to productivity, and to the extent that the postmodern cooperative tool-metaphor in computer design simply substitutes the earlier democratic-participative one, there will be no way to reintroduce in the analysis the lost collective dimensions of design which could prevent the coarse exploitation of people and their time. This may turn all talk about creativity into fashionable wishful thinking.

Promiscuous, interactive, communicative and busy togetherness coupled to selfish autonomy for the purposes of a creativity that is more a kind of physical productivity, leads the thoughts to what elsewhere has been called the "Don Juán syndrome" (Ivanov, 1986, p. 135, 139, 159). It has been rightly suggested (Heim, 1987, p.205f; Rifkin, 1987, p. 71, 179ff) that the frame of mind of the American educational system which has been exported to both the educational and working milieu of many other countries, places a premium on how fast we can recite an answer (from the "data base") or solve a problem. Pondering, reflecting, and musing might well be encouraged in certain cultures but the emphasis here is on keeping up, requiring compartmentalization, segmentation, quick absorption of material and even faster recall. The inner gestation of thought formulation is foreshortened. Ideational flow is emphasized over gestation, and what has been called "the active expectation of the not yet verbalized" grows shorter in span. There is an unwavering belief that intelligence and speed go together, i.e. that creativity is productivity, and that the bright learner is always the fastest learner (Ivanov, 1986, p.139, quoting Zbinden). This conviction is enhanced by interaction with the computer world where everything seems to be temporary and fleeting, and everything is subject to continual "creative" edits, revisions and modifications (Heim, 1987, p. 212; Lindfors, 1980, as quoted by Ivanov, 1986, p.130, both reformulate this very same conceptualization by Rifkin). Time loses its independent status as reference point and is now nothing but a resource. There is no well-established past, no preconceived future, but just an unceasing process of simulation. Realities change at such accelerated speeds that even scientific truths become impediments and ultimately become expendable. Consequently we witness the diffusion of both situational ethics and of the improvisational heuristics of degenerating research programmes, including even the "ability to suspend judgement in the face of disconfirming evidence", which will allow "proposals to be made without regret even when they have highly implausible aspects, or when tests are not likely to be possible in the foreseeable future" (Holton, 1984). This may be the background for some scientists' indignation with the praxis that is being formed in their fields (Chargaff, 1971). In the Don Juán style (as in Rifkin's descriptions), conditions change so fast that laws, contracts, and terms of agreement become shortlived. Realities are simulated, edited, and revised at lightning speed. Time being at premium, less and less is given over to making good on prior promises and more and more is given over to facilitating new options and agreements. Change is honored as the only timeless truth, and "to stay in business" in the hardening academic competition of the "higher capitalism" (Nisbet, 1971) is the yardstick by which the profitability of research interests is measured. Is this, perhaps, one reason for the increased interest for change, flexibility, interactivity, toolness, and creativity in the context of information systems research?

6. Conclusions

The surveyed literature suggests that most approaches to the issue of creativity have been empirical without any stronger theoretical basis for the conceptualization of creativity. Creativity has often been vaguely understood as a kind of dialectic integration between quantitative-formal-analytical-logical and qualitative-intuitive-imaginative thinking without any particular reference to e.g. psychological knowledge. The approaches have sometimes been subsumed under the less proper label of productivity or under the more appropriate label of design. The insights obtained from the most comprehensive studies of design of inquiring systems, however, have seldom been adduced or exploited in the applications to the computer field. The gap between the theorizing about creativity and the theorizing about software or computer systems development is sizable and seems to be growing. Recent studies of word processing and of the time phenomenon in the context of computer applications indicate that there is no reason for confidence about a coming improvement of the situation in view of the introduction of new computer technology. Organismic pantheism (as one may interprete Rifkin's vision) is one of the weak alternatives which have been lately proposed besides of feminism, communicative postmodernism, and the socialistic integration of idealism with positivism in the form of action research that recalls early pragmatism and the "Alliance intellectuelle franco-allemande" (Ivanov, 1986, p.119). The increased emphasis on research about change and creativity may itself be a symptom of the fact that research is influenced by the general trend towards provisional and superficial solutions of insufficiently understood problems. Such an interpretation is supported by the fact that valuable "old" approaches to the design problem , addressing questions that are not posed, are not recognized as relevant, and are not developed or applied to the issue of change and creativity.

7. Further Research

In the light of postmodernistic and high-tech developments, it is difficult to believe that creative better design of computer systems will depend upon the discovery or invention of smarter "methods" or "theories" for systems design or upon better preaching rhetorics. If people don't *care* and don't love their family or their neighbours, as present trends of divorces, crimes, and wars indicate, including academic wars, then the need and search for method may have to reside at another plane of inquiry. It has been pointed out (Churchman, 1987) that morality, in the classical literature, is presented as a state: either you sinned or you did not; either this policy or system is immoral, or some mixture of moral and immoral, or it is moral. Thus each act of a human is supposed to carry with it the idea of the state of morality of the act. Suppose now that morality is not a state of affairs, but a process, and that the process is design, of a just world, of international peace, or of a good computer system. A creatively good computer system, then, is a process of on-going struggle, an endless design beyond an ideal discourse, even if it's by no means obvious that it is a linear progression towards a better world. This ideal cannot be only a question of freedom. "Freedom from" bad things like poverty, bankruptcy, injustice, false data, etc. is only one part of the creative design. The other parts will be "freedom to", i.e. commited responsibility and engagement for how to use the surplus, the available true data, etc., in both an ethical and aesthetical way.

At this point become evident the limitations of the pragmatic approach, beyond its ability to point out the political, ethical, and, partly, the aesthetical aspects of scientific design (Churchman, 1979; Heim, 1987). It strives to reach beyond a rather sterile preaching for material plenty, co-operative progress and justice for future generations but it lacks the rhetorical power to move hearts (to account for love, von Baader might have said) and to connect the creative imagination of scientists to issues of art and religion as they should be relevant to everyday's small practical problems at work. This lack of rhetorical power, in the widest possible sense of rhetorics, reveals a perhaps unsourmountable gap between cognition and emotion conditioned by the heritage from the Kantian conceptualization of rationality which is shared by critical social theory, pragmatism, and mathematical-logical tools. In such a conceptualization of the world, nature no longer contains the slightest trace of mind and spirit. A complete science of nature is a mathematicized mechanics. Spirit, on the other hand, functions accordingly to completely different laws that are immanent to the spirit itself. "It follows that nature as the kingdom of necessity and the human world of spirit as the kingdom of freedom do not intersect" (Simmel, 1984, pp. 37, 48, there have been, however, excellent efforts to find an intersection in Churchman, 1971, chap. 10ff.). This is perhaps what fosters the unfortunate belief that practical life is one hundred times more important than theory or, for that matter, the other way around.

Obviously I have no answer or solution to this impasse, even if it is clear to me that certain systems schools originated by American pragmatism offer the best and most extensive bridging between natural science, formal science, and the human sciences. I have no faith in the various alternatives for creative thinking, Marxist, phenomenological, critical, postmodernistic or whatnot, which have been presented lately, and which do attempt to bridge the "hard" methodological issues of modern industrial technology and modern science to the eternal issues of morality and religion. I reject activism and eclecticism as well as the "chaotic multiplicity of cultural forms" representing a playful postmodern polytheism of values that tends to relativism and further to nihilism (Simmel, p.16). "While we believe that good is something to be invented, we demand of our rulers [and now also from our systems analysts, my addendum] such qualities as 'vision', 'dynamism', 'creativity', and the like. If we returned to the objective view [of the good] we should demand qualities much rarer, and much more beneficial -virtue, knowledge, diligence and skill. 'Vision' is for sale, or claims to be for sale, everywhere. But give me a man who will do a day's work for a day's pay, who will refuse bribes, who will not make up his facts, and who has learned his job." (Lewis, 1988).

In view of the "metaphysical" systems purpose of bridging the gaps of inquiry and of building up a conceptual scheme by means of dialogue it must be recognized that consistency or absence of explicit logical contradictions is obviously a negative criterion: it is necessary but not sufficient. Besides such low-level consistency one must look for "coherence", i.e. connection between fundamental principles, as well as applicability to *all* possible experience. The conceptual synthesis must show all experience to be interpreted without oversight, distortion, or "explaining away" on the basis of its key concepts (Ferré, 1987). Science as part of culture should account for the place that e.g. love, power, and the sacred have for the majority of the people in the world. Therein lies the challenge for future methods of systems development, and their ability to integrate work of the thinkers and traditions mentioned above.

One expectation I have is represented by some of the research ideas referenced in the above text. Another, and the main one, is to explore further in the near future some avenues of research by relating mathematical-logical play to the understanding of symbols, myths and advanced games or rites in the body of analytical psychology. *It is a question of the relation of the designing ego to the "self" in the individuation process, but also, or mainly, a question of seeing creativity as the psyche's integration of unconscious contents through the process of psychological "individuation"* (Ivanov, 1989; Jung, 1953-1979, see the general index in CW 20 under creative, creativeness and creativity). It is also, in other words, a question of regaining childlike innocence, play and spontaneity by giving up childish ignorance, games and unruliness (Hillman, 1971, p. 397, cf. also pp. 390, 399, 400f.).This may be seen as a better version of the critical social theory's "Selbreflexion" seen as interiorization of a therapeutic discourse or as an internal expression of the universal will of conversational consensus rooted in language (Ladmiral, 1984). Particularly fruitful could be to relate all this to modern pragmatism, experimental idealism and dialectical social systems theory, to concrete experiences of computer systems development, and further to the history of ideas where some of the above mentioned issues are seriously considered.

One main purpose of this paper was, then, to strive for a synthesis of some interrelated ongoing research efforts in order to foster researchers' mutual interest and help. One recurrent axiomatic or dogmatic theme will then be the question of whether the obtained insights really and "concretely" help somebody to develop better computer systems.

8. References

Ackoff, R. L., & Vergara, E. (1981). Creativity and planning: A review. *European Journal of Operations Research, 7,* 1-12.

Amabile, T., M. (1983). *The social psychology of creativity* . New York: Springer Verlag.

Atkinson, C. J., & Checkland, P. B. (1988). Extending the metaphor 'system'. *Human Relations, 41*(10), 709-725.

Bannon, L., & Schmidt, K. (1989). CSCW: Four characters in search of a context. *Proc. of the First European Conference on Computer Supported Cooperative Work, London, 12-15 September 1989* (pp. 358-372).

Barron, D., & Taylor, C. (1975). *Scientific creativity* . Krieger.

Blumenberg, H. (1985). *The legitimacy of the modern age* . Cambridge: MIT Press. (Originally published as *Die legitimität der Neuzeit.* Frankfurt: Suhrkamp Verlag, 1966, 1976.)

Bond, A. H., & Gasser, L., (Eds.). (1988). *Distributed artificial intelligence* . San Mateo, CA.: Morgan Kaufman. (With bibliography.)

Borning, A. (1987). Computer system reliability and nuclear war. *Communications of the ACM, 30*(2, February), 112-131. (With a bibliography of 142 entries.)

Bortoft, H. (1986). *Goethe's scientific consciousness (ICR Monograph No.22, ISBN 0904 674 10X)* . Cambridge, Wells (Kent): Institute for Cultural Research.

Brecht, A. (1941). The myth of IS and OUGHT. *Harvard Law Review, 54,* 811-831.

Buckley, M. J. (1987). *At the origins of modern atheism* . New Haven and London: Yale University Press.

Buttimer, A., (Ed.). (1983). *Creativity and context: A seminar report* . Lund: University of Lund, Dept of Geography, & CWK Gleerup. (ISBN 91-40-04896-9. With bibliography of 151 entries.)

Chandrasekhar, S. (1987). *Truth and beauty: Aesthetics and motivations in science* . Chicago and London: University of Chicago Press.

Chargaff, E. (1971). Preface to a grammar of biology: A hundred years of nucleic acid research. *Science, 172*(3984, 14 May), 637-642. (Orig. in *Experientia,* 1970, 26, No. 810.)

Churchman, C. W. (1948). *Theory of experimental inference* . New York: Macmillan.

Churchman, C. W. (1961). *Prediction and optimal decision: Philosophical issues of a science of values* . Englewood Cliffs: Prentice-Hall.

Churchman, C. W. (1971). *The design of inquiring systems: Basic principles of systems and organization* . New York: Basic Books.

Churchman, C. W. (1979). *The systems approach and its enemies* . New York: Basic Books.

Churchman, C. W. (1987). *The philosophy of design* (Unpublished manuscript). University of California, Center for research in management science, Barrows Hall, Berkeley, CA 94720, USA.

Cranberg, L. (1968). Law: scientific and juridical. *American Scientist, 56*(3), 244-253.

de Bono, E. (1970). *Lateral thinking: Creativity step by step* . New York: Harper & Row.

Ehn, P. (1988). *Work-oriented design of computer artifacts. (Doctoral diss.)* . Umeå-Stockholm: University of Umeå, Arbetslivscentrum and Almqvist & Wiksell International.

Ekvall, G. (1988). *Förnyelse och friktion* . Stockholm: Natur och Kultur.

Ferré, F. (1987). *Language, logic, and God* . Chicago & London: University of Chicago Press. (Reprint of the 1961 ed.)

Forsgren, O. (1988). *Samskapande datortillämpningar [Constructive computer applications]* (Doctoral diss., Report UMADP-RRIPCS-3.88). University of Umeå, Inst. of Information Processing.

Forsgren, O. (1989). A prototype of a learning co-constructor. *Proc. of the ISSS Int. Society for the Systems Sciences, 33rd Annual Conference, Edinburgh, Scotland, 2-7 July 1989. Vol 1* (pp. 159-164, 92-97).

Ghiselin, B. (1952). *The creative process* . New York: Mentor Books.

Gilchrist, M. (1972). *The psychology of creativity* . Melbourne: Melbourne University Press.

Goldkuhl, G., & Röstlinger, A. (1988). *Förändringsanalys: Arbetsmetodik och förhållningssätt för goda förändringsbeslut*. Lund: Studentlitteratur.

Gough, H. G., & Woodworth, D. W. (1960). Stylistic variations among professional research scientists. *Journal of Psychology, 49*, 87-98.

Guillet de Monthoux, P. (1981). *Doktor Kant och den oekonomiska rationaliseringen. Om det normativas betydelse för företagens, industrins och teknologins ekonomi*. Gothenburg: Korpen. (German trans.: Vulgärkantianische Unternehmenlehre. München: Leudemann, 1981.)

Harman, W. (1984). *Higher creativity: Liberating the unconscious for breakthrough insights*. New York: Jeremy P. Tarcher.

Heim, M. (1987). *Electric language: A philosophical study of word processing*. New Haven and London: Yale University Press.

Hill, R. C. (1989). Freeing the internal environment for problem-solving and creativity: A precursor to education and the future. *Proc. of the ISSS Int. Society for the Systems Sciences, 33rd Annual Conference, Edinburgh, Scotland, 2-7 July 1989. Vol. 1* (pp. 152-158).

Hillman, J. (1971). Abandoning the child. *Eranos, 40*, 357-407.

Holton, G. (1984). Do scientists need a philosophy? *Times Literary Supplement,* (November 2nd), pp.1231-1232.

Ivanov, K. (1972). *Quality-control of information: On the concept of accuracy of information in data banks and in management information systems* (Doctoral thesis). The University of Stockholm and The Royal Institute of Technology. (NTIS No. PB-219297.)

Ivanov, K. (1986). *Systemutveckling och rättssäkerhet : Om statsförvaltningens datorisering och de långsiktiga konsekvenserna för enskilda och företag [Systems development and rule of law]*. Stockholm: SAF:s Förlag.

Ivanov, K. (1987). Rule of law in information systems research: The role of ethics in knowledge-building procedures, especially in the updating of inference networks. In P. Järvinen (Ed.), *Proc. of the Tenth Information Systems Research Seminar in Scandinavia, Tampere-Vaskivesi, Aug.10-12 1987*. Tampere: University of Tampere.

Ivanov, K. (1988). Expert-support systems: The new technology and the old knowledge. *Systems Research, 5*(2), 293-100.

Ivanov, K. (1989). Computer applications and organizational disease. In C. W. Churchman (Ed.), *The well-being of organizations* (pp. 283-312). Salinas, CA: Intersystems. (Also as report LiU-IDA-ADB-R-83-2, University of Linköping, Dept. of Computer and Information Science.)

Johnson, P. (1987). *Moderna tider: Tjugotal till åttiotal*. Stockholm: Ratio. (Originally published as A history of the modern world: From 1917 to the 1980s, 1983.)

Jung, C. G. (1953-1979). *Collected Works - CW (20 volumes)*. Princeton: Princeton University Press. (R.F.C. Hull et al., Trans.)

Järvinen, P. (1985). Levels for utilization of computing. *Convention Informatique - Recueil des Conférences, 16-20 sept. 1985 (Tome A, ISBN 2-902574-18-5; ISSN 0376 494)*. Paris: SICOB Salon International d'Informatique, Télématique, Communication, Organisation du Bureau.

Järvinen, P. (1988). Human computer interaction research in the light of Aulin's foundations of mathematical system dynamics. *The workshop on human computer interaction and complex systems, Loch Lomond, Oct. 2-4, 1988*. (Draft available from the author, University of Tampere, Dept. of Computer Science.)

Kerola, P., & Taggart, W. (1982). Human information processing styles in the information systems development process. In J. Hagwood (Ed.), *Evolutionary information systems* (pp. 63-86). Amsterdam: North Holland.

Koestler, A. (1964). *The act of creation*. New York: Macmillan.

Ladmiral, J. R. (1984). J. Habermas. In D. Huisman (Ed.), *Dictionnaire des philosophes* (pp. 1123-1131). Paris: Presses Universitaires de France.

le Boeuf, M. (1980). *Du är kreativ*. Stockholm: Liber.

Lerda, F. (1988). *La creatività in matematica* (Unpublished manuscript). Università di Torino, Dept. of mathematics, Via Carlo Alberto 10, I-10123 Torino, Italy.

Levén, P., & Nordström, T. (1989). *Socially responsive systems* (Presented at the conference Support, society and culture: Mutual uses of cybernetics and science, The Int. Federation for Cybernetics, Amsterdam, March 27-31, 1989).

Lewis, C. S. (1988). *Christian reflections*. Glasgow: Collins. (Walter Hooper, Ed. First published in 1967.)

Lindfors, P. (1980). Ta ifrån mig mitt ansvar. *Jakobs Stege, 3*, 2-6.

Lyytinen, K. (1986). *Information systems development as social action: framework and critical implications* (Doctoral diss.). University of Jyväskylä, Finland, Dept. of computer science.

Mahoney, M. J. (1979). Psychology of the scientist: An evaluative review. *Social Studies of Science, 9*, 349-375.

Mathiassen, L., & Nielsen, P. A. (1988). *Soft systems and hard contradictions. Approaching the reality of information systems in organizations* (Unpublished manuscript). Aalborg, Denmark: Aalborg University Center, Dept. of Electronic Systems.

Mathiassen, L., Rolskov, B., & Vedel, E. (1983). Regulating the use of EDP by law and agreement. In U. Briefs, C. Ciborra, & L. Schneider (Ed.), *Systems design for, with, and by the users* (pp. 251-264). Amsterdam: North Holland.

Nagel, S. (1988). Decision-aiding software for all fields of knowledge. *Int. Journal of Information Management, 8*(2), 123-140.

Nilsson, K. (1987). *Project description: Design of interactive information systems* (Report UMADP-RRIPCS-5.87, ISSN 0282-0579). Inst. for Information Processing, University of Umeå, Inst. of Information Processing.

Nilsson, K. (1989). Designing for creativity: Toward a theoretical basis for the design of interactive information systems. *Proc. of the 12th IRIS Conference - Information Systems Research in Scandinavia,13-16 August 1989, Skagen, Denmark* . Aalborg: Aalborg University, Inst. of Electronic Systems.

Nisbet, R. (1971). *The degradation of the academic dogma. The university in America, 1945-1970* . London: Heineman.

Nissen, H. E. (1976). *On interpreting services rendered by specific computer applications* (Doctoral dissertation). Stockholm: The Royal Institute of Technology.

Nissen, H. E., et al. (1988). *AMIS: Ansvars- och medverkansformer i kontinuerlig systemutveckling* (Project description). Lunds University, Inst. of Information Processing.

Nurminen, M. (1988). *People or computers: Three ways of looking at information systems* . Lund and London: Studentlitteratur and Chartwell-Bratt.

Ottmann, H. (1982). Cognitive interests and self-reflection. In J. B. Thompson, & D. Held (Ed.), *Habermas: Critical debates* (pp. 79-97). Cambridge: MIT Press.

Papert, S. (1980). *Mind storms: Children, computers, and powerful ideas* . New York: Basic Books.

Pearl, J. (1984). *Heuristics: Intelligent search for computer problem solving* . Reading, Mass.: Addison-Wesley.

Peat, F. D. (1984). *Synchronicity: The bridge between mind and matter* . New York: Bantam Books.

Pirsig, R. (1974). *Zen and the art of motorcycle maintenance* . New York: Bantam Books.

Rifkin, J. (1987). *Time wars: The primary conflict in human history* . New York: Simon & Schuster.

Riley, P. (1986). *The general will before Rousseau: The transformation of the divine into the civic* . Princeton: Princeton University Press.

Rorty, R., (Ed.). (1967). *The linguistic turn: Recent essays in philosophical method* . Chicago: University of Chicago Press.

Scandura, J. M. (1980). Theoretical foundations of instructions: A systems alternative to cognitive psychology. *J. of Structural Learning, 4*, 347-393.

Schumacher, E. F. (1987). *A guide for the perplexed* . London: Sphere Books - Abacus.

Simmel, G. (1984). *On women, sexuality, and love* . New Haven: Yale University Press. (G. Oakes, Ed. and trans. Originally published in 1902-1922.)

Smith, G. J. W. (1986). *Upplevande och verklighet* . Lund: Studentlitteratur.

Smith, G. J. W. (1987). *Framtidens psykologi* . Lund university, Dept. of Psychology. (Avskedsföreläsning.)

Spengler, O. (1981-1983/1918). *The decline of the West - (2 Vols.)* . New York: A. Knopf. (C.F.Atkinson, Trans. Originally published, 1918. German ed. in München: Deutscher Taschenbuch, 1983.)

Stolterman, E. (1989). System design methods as creativity "killers". *Proc. of the 12th IRIS Conference - Information Systems Research in Scandinavia,13-16 August 1989, Skagen, Denmark* . Aalborg: Aalborg University, Inst. of Electronic Systems.

Strzalecki, A. (1989). Creativity and the styles of solving problems. *Proc. of the ISSS Int. Society for the Systems Sciences, 33rd Annual Conference, Edinburgh, Scotland, 2-7 July 1989. Vol. 4* (pp. 248-253).

Sørgaard, P. (1988). *A discussion of computer supported cooperative work* (Doctoral dissertation). Aarhus University, Dept of Computer Science.

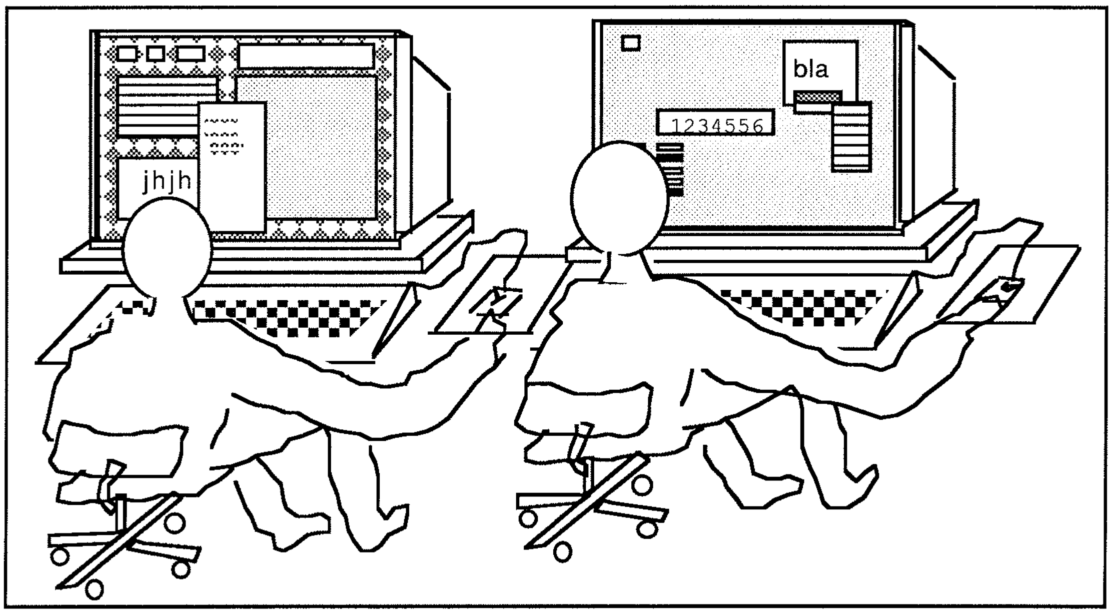
Ulrich, W. (1983). *Critical heuristic of social planning* . Bern: Paul Haupt.

Whitaker, R., Östberg, O., & Essler, U. (1989). Communications and coordination of concerted activity. *Human Interface News and Report, 4*(4), 325-338.

Östberg, O., Whitaker, R., & Amick III, B. (1988). *Den automatiserade experten. En uppsats om expertsystem och några intryck från en AI-konferens. (TELDOK technical report; overview in English: The automated expert:. Technical, human and organizational considerations in expert systems applications.)* . Stockholm-Farsta, Sweden: Swedish Telecommunications Administration.

A THEORY OF REFLECTED CREATIVITY WITHIN THE FRAMEWORK OF ONLINE PROTOTYPING

Niels Jacobsen
Department of Information and Media Science
University of Århus



Socrates: "The only thing I know is that I know nothing."

Abstract: Systems development is basically a creative process in which the user's praxis is changed. By developing a new tool for the user, his praxis is changed, because by using the new tool, he is performing the work in a different way. At one level of the user's understanding it is still the same piece of work, because the purpose at some level is unchanged. But by using the new tool, some of the sub-purposes of the work are changed. The systems design activity is a part of the systems development concerning the elaboration of the new tool and the changes of the user's praxis. The design activity implies both the creation of a new tool - a new being thing - and the process of learning the new tool in context of the user's work. The interaction between these two processes is what I will define as the creativity in systems development. To elaborate this concept I will first present a theoretical basis for the talking about being and learning. The concept of reflection is then introduced and discussed in context of the systems development process. On the basis of this discussion the main principles of the design of a design-tool are outlined, and the qualifications of the developer and the user, performing the systems development process, are drawn up.

Introduction

After my reading of "Understanding Computers and Cognition - a New Foundation for Design"[1] I found it interesting to elaborate a little more on the Heideggerian ontology. I think that we, in the seeking of a new foundation for design, most avoid the danger of eclecticism. It is important to re-formulate the adopted theories in order to establish a consistent theoretical basis for systems development. Thus the following is not an introduction to Heidegger's concept of being, but an attempt to introduce the ontological conception into systems development.

Being and Learning

The following theory of being and learning is based on inspiration from "Sein und Zeit" by Martin Heidegger[2]. The theory is explained in a series of steps, explained by examples from daily life.

1. One of the basic characteristics of Man is that he is always engaged in doing something.

   - Even when we sleep. We are then engaged in sleeping.

2. While thus engaged, Man uses the things by which he is surrounded.

   - Bread, air, beds, pencils and hammers.

3. We *understand* the things we use in terms of the purpose for which we use them, and things cannot be understood in any other way. The understanding of things is linked to the purpose for which they are used. We are unable to understand things until we use them.

   - I understand the fork as something to use for eating. I understand it because I use it. While opening a beer I maybe understand the fork as a bottle opener, and as something to clean while washing the dishes.

4. When we use things, the essence of their existence is the purpose for which they are used. Only things we can use exist for os. This means that things do not exist independently. Their existence is connected to the subject that uses them, and they only exist for this subject.

[1] [Flores & Winograd]

[2] [Heidegger]

- The fork has existence for me, as a tool for eating, and the bed has existence for me as a tool for sleeping.

5. The things that we can use for particular purposes are called *primary beings*. The thing, as a primary being, is defined - in fundamental terms - as our purpose in using it.

- My bicycle is a primary being for me as a tool for going to work. The bus is a primary being for me as something I use when it is raining, or my bicycle is punctured, or it is too far to go by bicycle.

- The primary being of the bus is determined by my purpose in using it. The fact that somebody else uses it for a different purpose is none of my business. For them it is an other primary being. The primary being of the bus is not connected to the bus, but just connected to the person who uses it.

6.Things are *secondary beings* for us when we are unable to use them for the purpose for which we intended to use them. Secondary being is the opposite of primary being, although this does not mean that secondary being is the same as neutral non-being. The secondary being thing is still connected to the subject, and is still determined by the intended use of the thing. A thing is only "non-being" for us when there is a particular purpose for which they cannot be used.

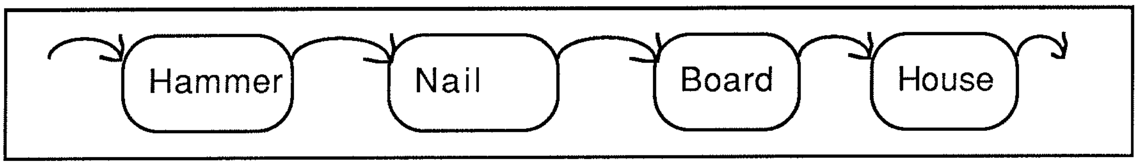
- My bicycle is a secondary being when I get up in the morning and discover that it is raining.

7. The purpose of what we do is never more than a sub-purpose.

- The purpose of hammering is to drive in a nail. And the purpose of driving in a nail is to fix a board. And the purpose of that is to build a house. Etc.

8. The fact that things are defined by their purpose is transitive. Tools refer to the job for which they are used, and the product of the job is a tool for an other job. Etc.

- See the figure. The hammer is understood through a chain of references. Each link in the chain is understood through the purpose, which refers to the next link.



The chain of understanding

9. The chain of references from the one partial purpose to the other, ends by the living of Man. In other words: life is the last purpose of everything.

- This statement is quite philosophical. I do not use it, but it has to be mentioned, because it completes the seeming infinite chain of references.

10. To *acquire understanding* of a thing is to *learn* how to use it for a certain purpose.

- I understood the mouse on my Macintosh when I learned to click on icons, point on menus and move windows with it.

11. During the learning, the thing you use can shift between having primary being and secondary being. If it serves your purpose it is a primary being, and if it does not serve your purpose it is a secondary being.

- When I click on the menu I wish to choose, I understand the mouse as a pointing device. It is a primary being. But if I miss the menu the mouse is no longer understood as a pointing device. It is a secondary being.

12. The shift from primary to secondary being indicates a *breakdown* in the understanding of a thing.

- Even when I try to reflect what a mouse is, my understanding of it breaks down. It is no longer a pointing device, but a thing which I think about.

13. When the breakdown has occurred, one of the links in the chain of purposes stands out - exactly that purpose which it does not serve. The breakdown happens at a certain *level of understanding*.

- This could for instance either be the understanding of the mouse as a tool for clicking icons, or as a tool for moving the cursor on the screen.

14. The shift from secondary being to primary being, is the indication of a learning.

15. A learning is a *reflected learning* if it happens in consequence of a breakdown in understanding. It depends on a reflection on how to use the tool in a different way.

- In a reflected learning you think about the thing in terms of the level in understanding in which the breakdown occurred. You try, for instance, to analyse the characteristics of the thing that makes it unusable for the purpose which became visible through the breakdown.

16. A learning is *unreflected* if we improve the ability to use the thing without any breakdown in our understanding of it.

- This is often the case when we learn movements. In this case we can learn how to do things more precisely without thinking of what we do.

17. A breakdown in the understanding of a thing might indicate that we change the purpose for which we use it. It becomes an other primary being.

- When Archimedes stepped into his bathtub, the water ran over. This caused a breakdown in his understanding of the water as something to use for bathing. He started thinking of the water running over, and suddenly he understood that it could be used for another purpose: He had found the law of Archimedes! (Buoyancy is equal to the weight of the displaced water.)

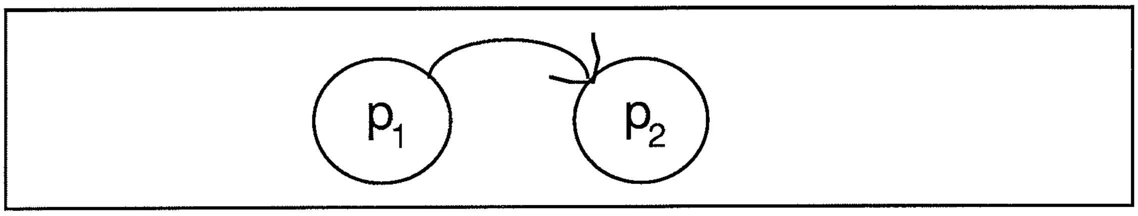
The creativity

The starting point for the creative process of designing a new tool, is the user's praxis so far; that means the user's understanding of the tools used while working. The undesigned tool does not exist at all, neither as primary being nor as secondary being.

The creation of the new tool implies that it becomes a primary being thing for the user, - it becomes a tool for him. The creative process in designing is therefore a process of learning, too. Contrary, the learning to use a new tool is in itself a creative process. Through the learning is obtained an understanding of a new thing; there is created a new primary being thing for the user.

The reason why I, in the description of the creative process, distinguish between the creation and the learning, is that the two concepts focus on different elements of the design process. The creation emphasizes the element of being presented to something new. This implies in praxis that the developer must come up with some hardware and software. In that way the developer is an active part of the creative process. On the other hand the learning emphasizes that it is the use (the learning) of the new tool, that changes the user's praxis.

As well as the learning, the creative process can be either reflected or unreflected. The unreflected creativity consist in a change of praxis without any breakdown in the understanding of praxis. That is to say that you have not been thinking of the change you have made (see the figure next page).

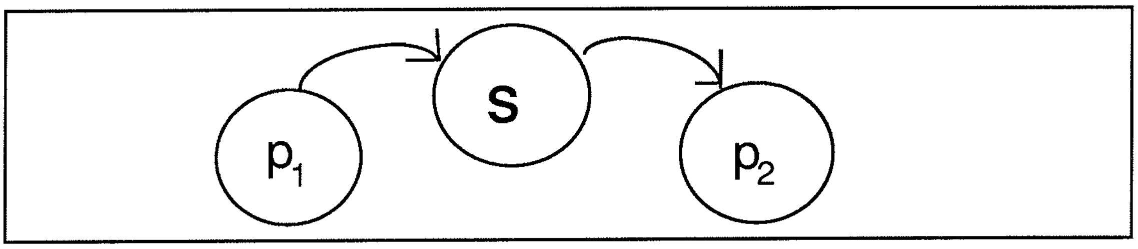


The unreflected creativity

The new understanding, P2, of the tool replaces the old understanding, P1, without any breakdown in the understanding. Thus the tool remains being a primary being through the process. The exploitation of unreflected creativity presupposes that the user is experienced in changing his own praxis[1], and that he has the possibility of changing the tool himself.

In the situation where the users don't have much experience in changing there praxis related to the use of computers, I think the design process most be built on the reflected creativity.

The reflected creativity is characterized by having its starting point on a breakdown in the users understanding of the work or the tool being created. The change in the user's praxis caused by the new tool, happens in the light of reflections on the work. This reflection presupposes a breakdown in the understanding of the tool, in which the user reflects the level of understanding in which the breakdown has occurred. (See the figure.)



The reflected creativity

Before the breakdown the tool is a primary being, P1. After the breakdown it is a secondary being, S. The secondary being tool is an object for reflections that cause a change of the tool. But now it is understood (used) in a new way; and therefore this is an other primary being, P2.

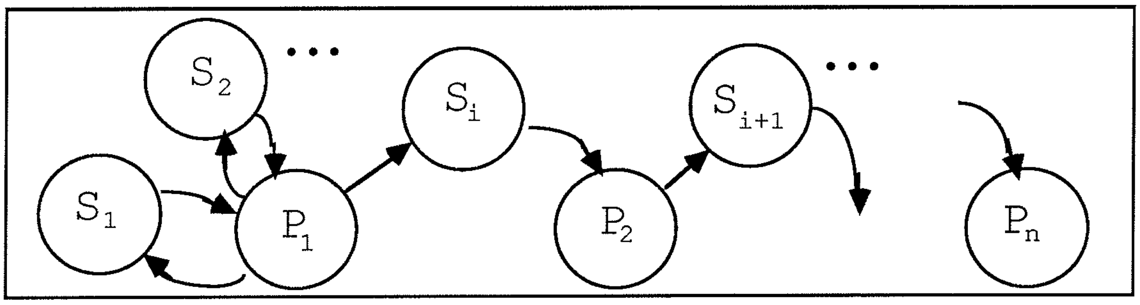
[1] Cf [Dreyfus & Dreyfus]: Only the expert can make the right decision intuitively.

How to create the breakdown in understanding

Now the matter of discussion is how to create the breakdown in the users understanding. One possibility is to provoke the user to think about his own work. This indicates in itself a breakdown in the understanding of the work. But the level of understanding is determined by the matter the user began thinking about. This can be used in the design process, if the developer confronts the user with an assertion or asks a question about the work. The user then begins thinking about his work on the basis of an interpretation of the developer's assertion. This method might be useful under exceptional circumstances, but the drawback is that it is the developer's interpretation of the user's work that will guide the creativity. It is a point, too, that the breakdown in itself does not indicate any creativity with respect to the design, because there is not created any new tool for the user. But this kind of breakdowns can be used creatively in the creation of a description of the user's work. The description could then be understood as a tool for designing the user's new tool. Not until the user starts using the tool being designed, there is a creative process with respect to the tool. Not until that moment the new tool is a primary being for the user.

Example 1

In the analysis of a traditional systems development process[1], the creative process has as its starting point a series of breakdowns in the user's understanding provoked by the developer's questions. On the basis of his reflections on these breakdowns the user makes some statements of his work. The developer uses these statements in creating a formal description (eg flow-charts). The developer understands this description as a tool for making the users new tool. But the creative process does not start until the user starts using the new tool (see the figure below).



The creative process in a traditional systems development process

All the breakdowns provoked by the developer are breakdowns in the user's understanding of the old tool/praxis, ie the primary being tool, P1. Each of the breakdowns causes the tool to be secondary beings, S1-Si, for the user, who then thinks about the

[1] Eg [Yourdon]

use. But the reflections do not imply any learning, because either the tool or the use of it is changed. Not until the developer has developed the new tool, the creative process does begin. The creative process then is formed as a usual learning process in which the tool's primary being changes, P2-Pn, in correspondence with the successful learning.

This use of breakdowns is characterized by breaking down the user's understanding on a very detailed level; and by not being followed by a learning.

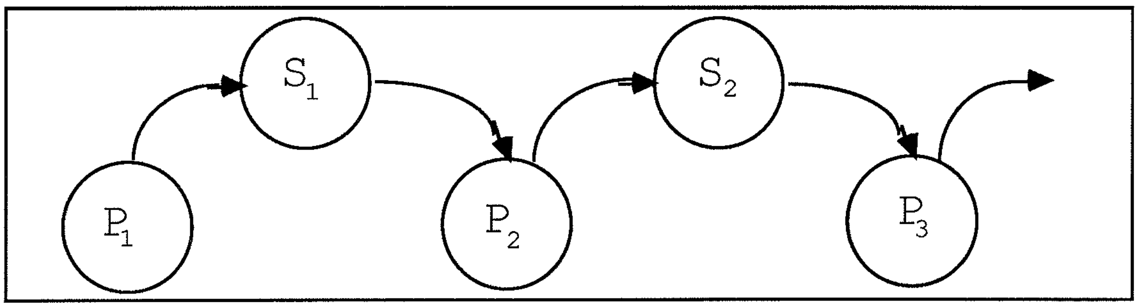
Example 2
An other possibility is to provoke the breakdowns on a very high level of the users understanding. The use of metaphors[1] in the design process is an example. The user is provoked to think about his work in terms of an other (well-known) working process or tool. In this case, too, the breakdowns are not used directly in the creative process. But they might be useful indirectly, because the later learning of the new tool can be more easy. When the tool has been developed, the metaphors, used by the developer in creating the breakdowns, now can be used as a reference in the user's reflected learning of the tool. Furthermore the metaphors can make an unreflected learning possible if the tool can be understood without breakdown in the understanding.

The successive reflected creativity

The technique I will propose is quite different. I will make the breakdowns spring up during the use of the tool. The creative process hereby takes it's start in the properties of the tool which are not sufficient. The problems are, however, how to use a tool not developed, and how to use the breakdowns in a creative manner.

I think this could be the case in using prototypes. By giving the user a prototype of the tool, the developer can provoke a breakdown in the user's understanding of the tool on exactly the level of understanding on which the tool does not fit the use. The breakdown often can be read in the users reactions: he stops the work and tells verbally or by bodylanguage that something is wrong. The developer's interpretation of the breakdown is used in developing a new prototype which is given to the user. This goes on in a successive process (see the figure next page).

[1] Cf [Halskov]



The creative process in prototyping

The tool is in this way developed in an interplay with the creative process, because each prototype becomes a primary being for the user, and each breakdown is followed by a reflection, which is incorporated in the following prototype.

To increase the connection between the user's creative process and the developer's developing, I think it is important to have as many as possible of these changes in understanding in the design process.

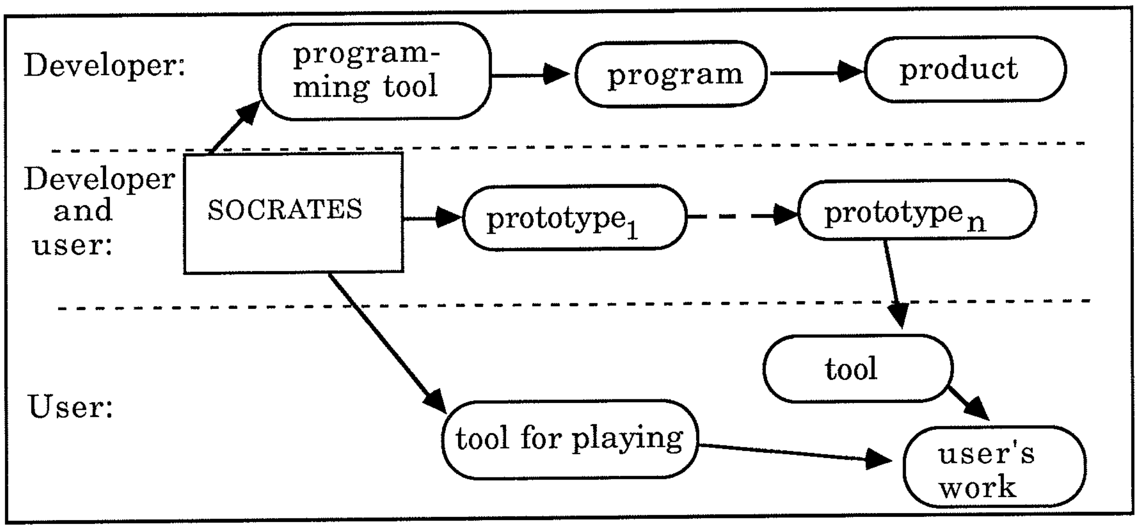
An example: SOCRATES a tool for design

To concrete the theory above I will make an example of the use of it. In doing this I have to deal with the problems of organisation and communication. I have chosen to concentrate on the communication between the developer and the user, and eliminate all other aspects (although they are very important in real-world situations). Thus the following discussion of a tool for design takes place in an idealised process of prototyping in which there are only one developer and one user. My point is to illustrate the theory, and not to discuss the benefits of prototyping!

To exemplify the theory I will sketch some principles of the design of a tool to be used in the process of reflected creativity in design. The tool, called SOCRATES, is a tool for making horizontal prototypes.[1] The use of breakdowns as described above implies that the user is confronted with something simulating the future tool.. It is necessary that the simulation can be understood and used as if it was a tool. The tool has to be a primary being before the breakdown. Horizontal prototyping makes it possible to implement a simulation of the future tool, that at the same time can be understood by the user as a tool, and by the developer as a preliminary description of the system to be developed. To make it possible to use the breakdowns creatively it is important that the prototype can be changed quickly in an experimental process.

[1] Cf [Floyd p.4]: *The functions are not implemented in detail as required in the final system; thus they can be used for demonstration, part of their effect being omitted or simulated ("horizontal prototyping").*

SOCRATES is used in a series of meetings between the user and the developer, in which the design of the tool is improved successively. At first they do not know very much about the design of the tool, but the developer can teach the user how to interact a computer, and what can be done by means of it. As the user learns to interact the computer, they can develop the first prototype of the new tool. Through experiments, in which the user tries which changes are best, the prototype is successively improved. Each prototype is the basis for developing the next. (From the developer's point of view it is perhaps not improvements, because a rejected proposal by him can be interpreted as a step backwards.) Each prototype is in that way a tool for the developer and the user in an activity they both think is common to them. They both have the purpose to improve the prototype. While the user tries to use the prototype as a tool for doing his work, the developer is on the look-out for breakdowns in the user's understanding of the tool. These breakdowns are the starting point for reflections and discussions about what changes to be done next. In these discussions the developer and the user both understand SOCRATES as a tool for design. But this implies that they each can understand SOCRATES in an other way (see figure below).



The understanding of SOCRATES

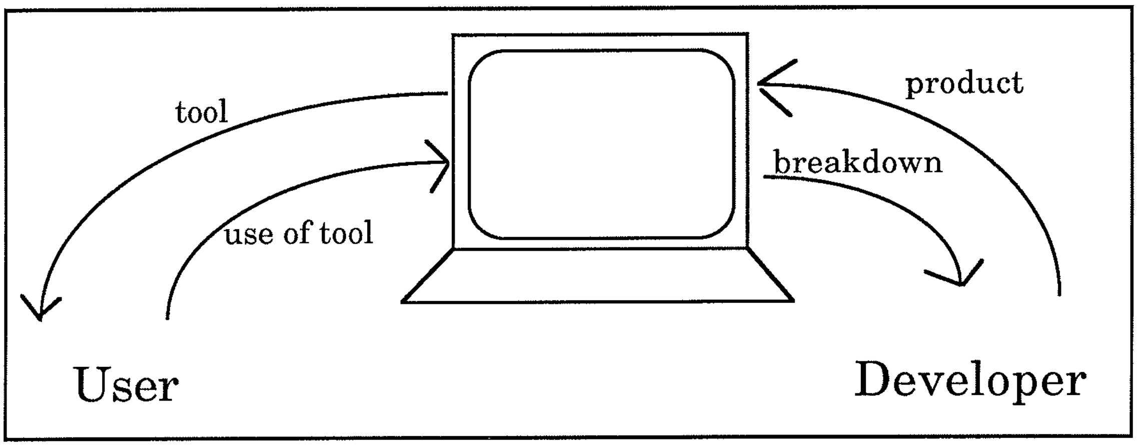
The user must be able to understand SOCRATES as a tool for doing his everyday work, because he has to decide if the design is satisfactory. At least in the beginning this claims some ability to abstract. The user has to "play" the work, and in his play use the prototype as if it was a real tool. In the play SOCRATES is understood in terms of the purpose of the user's everyday work. At the same time it is necessary that the developer can act as a programmer and understand SOCRATES as a tool for programming with the purpose of developing a product.

The user's understanding of SOCRATES as a tool for designing can influence his understanding of it as a tool for working. If he begins to understand it as a simulator, he can use it (play his work) in a way he would not have dared if it was a real tool working on the real world. This is important because the creativity can break down the limits of circumspection. Quite new ideas of the design can be created.

When the tool is designed, the developer and the user turn back to their usual work, and one could object that the joint design process is an illusion. The purpose of designing a tool is indeed common to them, but the final purposes are different. For me the point is, that it is sufficient to have an apparent common purpose. Not until the process is finished the different purposes become visible.

The design tool is media and reference

As sketched above SOCRATES is a design tool developed to increase the ability to use breakdowns as a key in the communication between the developer and the user. First of all SOCRATES is used as a media in the dialogue of questions and answers (see the figure).



SOCRATES as a media

The "questions" from the developer consists of a proposal for the design of (some part of) the tool. The users "answer" consists of the trying to use it. The question does only have a meaning in the dialogue if the user is able to behave like using the proposal.The meaning of the question <u>is</u> simply the use of the proposal. In the same way the answer only has a meaning in the dialogue if the developer is able to distinguish the user's successful use from a breakdown in his understanding of the tool. A breakdown indicates either that the user has to practice some more, or that the tool is not good yet (or perhaps the the hole setting has to be changed). On the basis of his interpretation of the user's breakdown the developer can make a new proposal as a new question. A question can either be an elaborating or a summarizing one. An elaborating question is a variant of the previous proposal, in which a breakdown is followed up by investigation of how to design the tool in a way to avoid the breakdown. A summarizing question is the presentation of the tool successfully designed so far, according to the developer.

Possible breakdowns in the user's understanding of this may correct the developer in his interpretation of what is well done so far.

In the dialogue SOCRATES is a media between the developer and the user, but their means of expression are different. The developer expresses himself through the development of a tool, and the user expresses himself through the breakdowns in his understanding (use) of the tool. Even a non-breakdown is an expression. It is a talking silence, telling that everything is OK so far.

An other important element of the communication is the verbal dialogue supplementing the non-verbal dialogue described above. The breakdowns in the user's understanding of the tool are starting points for discussions about how to improve the design. In these discussions SOCRATES acts as a reference in the talking about the tool. The physically appearance of menus, windows and devices is a visible support to the discussions; the user's actions and interaction can be referred.

In this way SOCRATES supports the development of common means of expression. They can both point at the screen, and develop concepts referring objects and actions. These shared concepts and means of expression are very important because they make a conversation possible even if the understanding of the actions referred are different.

Design of SOCRATES

For reasons which should be clear at this moment, I cannot give a description of SOCRATES which makes you understand it as a tool for designing and programming. But I will give a short description of the facilities of it as a product.

Two work-stations

The use of breakdowns in the design process implies that the user's understanding of SOCRATES, as a tool for doing his job, must be supported. At the same time SOCRATES has to be a good programming tool for the developer. And, what is very important, the developer's technical work must not disturb the user in using the prototype.

For these reasons SOCRATES is implemented on two work-stations (using NeWS on SUN work-stations). The one used by the user; the other by the developer. To support the mediation the two work-stations are connected in a way that makes it possible for the developer to "send" a proposal of design to the user's work-station.

The two work-stations are placed beside each other. This is important because it makes it possible for the developer to watch the user using the tool, and discuss the design with him.

On-line prototyping

The understanding of SOCRATES as a design tool implies that each prototype is understood as a tool for making the next. To support this understanding it is necessary that changes on the prototype can be made and tried quickly. The changes have to be implemented while the user continuously uses the prototype as a tool for working. This is made possible by implementing SOCRATES in an object oriented way, which supports replacement of instances and classes in a running system. (Unchanged variables and processes are automatically transferred, and sub-classes and their instances are automatically updated.)

Watching facilities

To increase the developer's possibility to watch the user's interaction on the tool, SOCRATES has some simple facilities implemented.

- Journaling, which makes it possible to "record" and "play" the user's interactions for a period.

- History-recording, which makes it possible for the developer to record relevant comments from the objects interacted by the user or by himself. The history is showed in a window on the developer's work-station, and recorded on a file as well.

- Screen-print (full-scale laserprint) is implemented to freeze a situation for later use.

Skill for design

The suggested use of breakdowns in the design requires some new skill of the user and the developer. The most important aspect of this is the ability to co-operate during the design process. For this reason the initial learning takes place in common.

User skill

The user must learn how to operate the computer. This initial learning most include different kinds of interaction, keyboard, mouse etc. The advantages and disadvantages of WYSIWIG[1] and YAFIYGY[2] must be illustrated as well. This can be done by showing the user some easy-learned applications or games, and maybe some applications concerning the user's daily work.

An other important thing is to teach the user the essential use of prototypes. This can, for instance, be learned by using paintings instead of computer applications: The user

1 WYSIWIG = What You See Is What You Get

2 YAFIYGI = You Ask For It You Get It

must "dictate" a painting to the developer without showing it to him. The developer's first draft or interpretation is then xeroxed and showed to the user. Knowing the developer's interpretation of the dictation the user then tries to give a better description, and so on. This "play" can be used to show the user that design is a successive process, and that the developer has to be corrected again and again because he does not understand.

The user furthermore must have the ability to play his work. This is necessary because he has to use the prototype as if it was a real tool. Every child knows how to play, and during the play all requisites are real; they are primary beings to the child. But most adult people have not maintained this skill. To revive this ability it is a good idea to make the user show (some parts of) his work by means of the present requisites. This play has an other important point; namely to show the developer some properties of the work. Of course the play does not replace an ordinary analysis, but is supplementing it. Furthermore the user by this play is trained in talking about his job.

Developer skill
During the prototyping - the settings - the user's most important skill of course is the ability to do his job (to play it). The developer must be experienced in doing his job, as well. This implies at least four roles to be played by the developer:

- the programmer
- the stage director
- the conductor
- the co-designer

Programmer
To play the role of the programmer, the developer, besides having the traditional abilities of programming, must be experienced in on-line programming. He must be able to make the changes of the prototype quickly during the settings.

The developer acts as a programmer between the settings, as well, by implementing the changes not implemented during the settings, and by preparing the prototype for further changes.

Stage director
By the role of the stage director I mean the role of building up the settings of the design process. Depending on the user's ability to play, the developer must create the necessary illusions, partly by means of the design tool and other physical requisites, partly by means of creating the user's expectations.

The role of the stage director is mainly played before each setting, but if the user's understanding of what is going on during the setting is breaking down, the developer must take up the role as stage director to keep the "show" going on. The ability to make a good analysis must be a part of the developer's skill, because he has to know as much of the user's work as possible.

Conductor

Playing the role of the conductor the developer must lead the experiment. The breakdown of the user's understanding must take place according to a superior plan of what parts of the tool to be designed; but in handling each breakdown he must act intuitively. This requires that the developer is able to distinguish the levels of the user's understanding of his tool. He must be able to "read" the breakdowns, because the user's expectations and understanding of the tool are expressed through them.

This role is played all during the settings. The developer must unremittingly be out of the touch with the situation: What is going on now? What to be done next? Is it time for coffee? Etc.

Co-designer

The role of the co-designer primary claims that the developer concretes the user's articulated and un-articulated ideas of the future tool. But the ideas most be born as well. This claims that the developer inspires and motivates the user. This is mainly done by presenting for the user the consequences of his ideas (like Socrates does in Plato's dialogues[1]). But the co-designer also takes care of the concepts and the references in discussing the design.

If the design process is breaking down the developer must leave the role as co-designer and let the stage-director take over to re-build the setting.

[1] This is why the prototyping tool is called SOCRATES.

Acknowledgment

This paper is based on my master thesis written together with Jan Hauerslev. I want to thank him for the co-operation.

References

[Dreyfus & Dreyfus]	Hubert L. Dreyfus & Stuart E. Dreyfus "Mind over Machine" Free Press, New York, 1986
[Flores & Winograd]	Fernando Flores & Terry Winograd "Understanding Computers and Cognition" New Jersey, 1986
[Floyd]	Christiane Floyd "A Systematic Look at Prototyping" in R. Budde et al.(eds): "Approaches to Prototyping " Spinger Verlag, 1984
[Halskov]	Kim Halskov Madsen "Sprogbrug og design" PB-245, DAIMI, University of Aarhus, 1988
[Heidegger]	Martin Heidegger "Sein und Zeit" Tübingen, 1967
[Yourdon]	Edward Yourdon "Managing the System Life Cycle" Yourdon Press, New York, 1982

RANKING DOCUMENTS FOR RETRIEVAL BY MODELING A RELEVANCE DENSITY.

Yana Kane-Esrig, George Casella
Cornell University, Ithaca N. Y. USA
Lynn A. Streeter, Susan T. Dumais
Bellcore, Morristown N. J. USA

Abstract

A long standing problem in information retrieval is dealing with queries that are best answered by two or more distinct sets of documents. We propose a new document ranking method, the *relevance density method*, to address this problem. The method was tested using probability functions that are mixtures of bell-shaped components. These functions produce a multimodal relevance density over the space in which documents and keywords are represented by vectors.

When multimodal queries were constructed, the relevance density method with a single feedback step performed better than the comparable vector averaging technique. There was no performance penalty paid for using the new method when the query was unimodal. Computationally, the new method requires fewer operations than vector averaging. The savings in the number of operations increase as the size of the collection increases. In our tests the savings were a factor of ten in one collection and a factor of two in another.

We propose a new method of ranking documents in information retrieval. The method, referred to as the *relevance density method*, is designed to improve upon the commonly used method of ranking documents by being able to respond correctly to a query which must be answered by two or more disparate sets of documents. In addition, for a large and varied document collection, the proposed versions of the relevance ranking method require fewer steps than the commonly used method.

The relevance density method can be applied whenever terms and documents are represented by vectors (or points) in the same n-dimensional space, with similarity in content reflected by the closeness of the term and document vectors in that space. The method ranks the documents by constructing a probability density over the document-term space. The ranking is done using the height of the density over the points in the space corresponding to the documents in the collection.

In this paper we briefly discuss a commonly used method of ranking documents. We then describe the general formulation of the relevance density method, and finally present two examples of the method with some initial test results. Proofs of the mathematical results stated in the paper are available in [Kane-Esrig 90].

Introduction.

The task of an information retrieval system is to respond to a user's request for information, called *a query,* with a list of sources of the information relevant to the query. These sources, called *documents*, are texts, e.g. books articles etc., which contain either the desired information or descriptions of the persons or organisations that can provide the desired information.

An information retrieval system responds to a user's query by ranking the documents which are available to the system (its *document collection*), in the order of their estimated relevance to the query. Since the document collection is usually very large, the user is presented only with the top-ranking documents. A good ranking method should give high ranks to the documents which are indeed relevant to the current user's query.

I. Existing Methods of Ranking Documents.

Usually, the documents and the queries are indexed by *terms*, also called *keywords*. It is assumed that much of the content of a document or a query is reflected in the terms, e.g. words, phrases etc., used in the text of that document or query.

The frequency of occurence of ith term in jth document can be recorded as the ijth entry of a *term by document matrix*. On the basis of the information contained in this matrix documents and terms can be represented by vectors in the same multidimensional space, which we will call *the document-term space*. Similarity or relatedness of the content of the documents and/or terms is reflected by the closeness of the corresponding vectors in the space. Such a space can be constructed, for example, by applying a matrix decomposition technique, Singular Value Decomposition (SVD), to the term by document matrix [Furnas 88].

There are two common methods of ranking documents in response to a user's query, (1) vector averaging and (2) probabilistic retrieval [Van Rijsbergen 79] [Bookstein 82] [Losee 87]. Of these two, only vector averaging is commonly used in conjunction with representing documents and terms as vectors in the document-term space discussed above. We will briefly review the vector averaging method in this section.

I.1 Vector Averaging

Vector averaging [Salton 68], [Salton 83] represents a query by a single vector in the document-term space. This query vector is a weighted average of the term vectors corresponding to the terms used in the query. Documents in the collection are ranked by some measure of similarity between their vectors and the single vector representing the query. The cosine and dot product are used often as similarity measures.

To increase retrieval effectiveness, the system also can incorporate relevance feedback. After the documents are ranked on the basis of the original query, the user is presented with the top ranking documents. Next, the user marks each of these documents as relevant or not-relevant. A new query vector is then formed by taking a weighted average of the document vectors marked relevant. The documents are then re-ranked on the basis of their similarity to the new query vector. This feedback step can be repeated several times.

Averaging relevant term or document vectors into a single query vector is reasonable when the vectors of the relevant documents are clustered together in a single region of the document-term space. The query vector is the system's estimate of the center of that single relevant region of the space. However, if the vectors of the relevant documents are arranged in two or more clusters in the space, and these clusters are separated by regions containing non-relevant documents, then averaging will not perform well, since it will tend to retrieve documents from the middle ground between the clusters. In fact, documents in the middle might have no relevance at all to the query. This situation can arise when a query deals with two or more separate topics or when no single document can answer the query, but several documents, dealing with separate aspects of the query, would be adequate.

The problems associated with always representing the query with a single vector was recognized a long time ago, but did not receive much attention. Query splitting was proposed by [Borodin 68]. The proposed algorithm treats the original query in the same way as ordinary vector averaging. At the feedback stage, the documents marked relevant are clustered into one or more groups on the basis of the correlations of their vectors. A separate query vector is computed by using vector averaging in each such group.

Query splitting has not replaced the usual vector averaging method in information rerieval due to several drawbacks, the most serious of which is the increase in computing time. All the operations involved in processing a single query vector have to be repeated for each group of documents separately. In addition, several separate rankings of the document collection are formed and it is not clear how to combine them into a single ranking without performing more prohibitvely expensive calculations.

The paper describes the results of a small test of the query splitting method. The results of the test showed some improvement in precision at high levels of recall over the usual vector averaging method.

II. Relevance Density Method.

We treat relevance as a continuous quantity and model its distribution as a probability density $\prod\left(D|Q_0,..,Q_1\right)$ over the document-term space. The higher the value of this "relevance density" over a given document vector $\underset{\sim}{D}$, the more relevant the corresponding document.

II.1. Notation and explanatory comments:

We will start with the prior density $\prod_0$ and use Bayes rule to update the density as we get new information in the form of the original query and relevance feedback.

$$\prod_m\left(D|Q_0,...,Q_{m-1}\right) = f\left(Q_{m-1}|D,Q_0,...,Q_{m-2}\right)\prod_{m-1}\left(D|Q_0,..,Q_{m-3}\right)/f\left(Q_{m-1}\right) \quad (1$$

$\prod_m\left(D|Q_0,...,Q_{m-1}\right)$ is the relevance density after the mth stage of feedback. The exact shape of the density function is not important. What matters is the ranking of documents that the density assigns to the collection. The density should be high over the area(s) of the document space where the relevant documents vectors reside and lower over the non-relevant documents. If there is more than one separate area in the space where there are clusters of relevant documents' vectors, the density should be multimodal. If the relevant documents are tightly clustered and the distinction between relevant and non-relevant documents is sharp (as opposed to having some documents partially relevant), then the mode(s) should fall off rapidly.

$f\left(Q_{m-1}|D,Q_0,\ldots,Q_{m-2}\right)$ is the sampling function which will determine how the new information is incorporated into the density surface we are trying to construct.

Π_0 is the prior relevance density. Prior information about the user can reflect knowledge about the interests of the particular user, or it can reflect knowledge about the interests of a category (age, occupation etc) to which the user belongs. If no prior information about the user is available, Π_0 is uniform (constant) over the space.

Q_0 is the original query. It is the collection of terms used in that query.

Q_i $i>0$ is relevance feedback received at ith iteration of feedback. It consists of a set of documents presented to the user at the previous stage and the judgments (e.g. relevant or not-relevant) made by the user.

$f\left(Q_m\right)$ is the marginal density. It acts as a scaling constant which will not affect the ranking of the documents, so we will ignore it and not compute it.

To simplify the task we will ignore the documents marked not-relevant in the relevance feedback and treat relevance feedback Q_m similarly to the original query Q_o. The only difference is that $Q_o=\{T^{(1)},\ldots,T^{(k_o)}\}$ consists of terms specified by the user, while $Q_m=\{D^{(1)},\ldots,D^{(k_m)}\}$ consists of documents marked relevant by the user. Both terms and documents are represented by vectors in the same space. We will use the same form of $f\left(Q_m|D\right)$ for the original queries and for relevance feedback for the rest of this paper. To simplify the notation, we will conduct all the discussion in terms of $f\left(Q_o|D\right)$. The only change that is required is to replace $Q_o=\{T^{(1)},\ldots,T^{(k_o)}\}$ by $Q_m=\{D^{(1)},\ldots,D^{(k_m)}\}$.

II.2. Constructing the sampling function f().

We will proceed by deciding on a set of properties that (in our opinion) a reasonable sampling function should possess, proposing probability densities which possess those properties and testing retrieval effectiveness obtained when these densities are used in (1).

II.2.1 Desired properties of the sampling function f().

(i) We will start by specifying the desired behaviour in a simple special case. When the query consists of a single term, it seems reasonable to make $f\left(Q_o|D\right)$ unimodal and to rank documents in the order of similarity of their vectors to the vector of the query term. The similarity measure that we chose is the cosine between the term vector $\underset{\sim}{T}$ and document vector $\underset{\sim}{D}$. The cosine was chosen because it has reasonable theoretical properties [Jones 87], has behaved well in tests [Noreault 81], has been used in several existing systems, e.g SMART, Bellcore ADVISOR, and is easy to compute. By taking the cosine as the measure of similarity we are projecting all the term and document vectors on a unit sphere in the term-document space.

(ii) If all the vectors of the terms of the query are located within a single circle on the unit sphere, then the total probability of this circular area should be higher than the total probability of a circular area of the same size that does not contain any of the vectors of the terms of the query.

(iii) We want the function to be unimodal when the term vectors are arranged in a single cluster and multimodal when there are more than one clusters. While it is hard to specify exactly what one or several clusters in R^n look like, we can define the desired behavior in extreme cases. The extreme case of "one cluster" is the situation where all the terms in the query should be treated equally (i.e they have equal weights etc.) and the term vectors coincide, i.e $\underset{\sim}{T}^{(1)}=\underset{\sim}{T}^{(2)}...=\underset{\sim}{T}^{(k_o)}$. In this situation $f\left(Q_o|\underset{\sim}{D}\right)$ should be unimodal with the mode over $\underset{\sim}{T}^{(1)}$. The extreme case of "two clusters" is the situation where all the terms in the query should be treated equally and the term vectors are divided into two equal groups pointing in the opposite directions, i. e. $\underset{\sim}{T}^{(1)}=...=\underset{\sim}{T}^{(\frac{k_o}{2})}$ $\underset{\sim}{T}^{(\frac{k_o}{2}+1)}=...=\underset{\sim}{T}^{(k_o)}$, $\cos\left(\angle\left(\underset{\sim}{T}^{(1)},\underset{\sim}{T}^{(k_o)}\right)\right)= -1$. In this situation, $f\left(Q_o|\underset{\sim}{D}\right)$ should be bimodal with the modes over $\underset{\sim}{T}^{(1)}$ and $\underset{\sim}{T}^{(k_o)}$.

(iv) The density function should not be zero over the vector of any term of the query, (provided the term has non-zero vector).

II.2.2 Two probability densities which possess the desired properties.

We will consider two probability densities which comply with the list of desired properties outlined in the previous section. These densities are not the only ones possible and in further work we might consider other possibilities.

The two densities are :

1) $f_1\left(Q_o|D\right)=\sum_{j=1}^{k_o} w_j * c(b_j)\left(\cos\theta_j+1\right)^{b_j} \qquad \sum_{j=1}^{k_o} w_j = 1,\ b_j>0$

Notation:

* θ_j is the angle between the document vector and the vector of the jth term: $\theta_j=\angle\left(\underset{\sim}{D},\underset{\sim}{T}^{(j)}\right)$, $\cos\theta_j=\cos\left(\angle\left(\underset{\sim}{D},\underset{\sim}{T}^{(j)}\right)\right)=\frac{\underset{\sim}{D}\cdot\underset{\sim}{T}^{(j)}}{|\underset{\sim}{D}||\underset{\sim}{T}^{(j)}|}$.

* $c(b_j)=[\int_0^{\pi}\left(\cos\theta + 1\right)^{b_j}d\theta]^{-1}$ is the normalizing constant.

* w_j is the weight of the component due to $\underset{\sim}{T}^{(j)}$

* b_j is the parameter of concentration of the component due to $\underset{\sim}{T}^{(j)}$ (We will discuss the parameter of concentration later in this section)

2) $f_2\left(Q_o|D\right)=\sum_{j=1}^{k_o} w_j*c(b_j)\exp\left(b_j*\cos\theta_j\right) \qquad \sum_{j=1}^{k_o} w_j = 1,\ b_j>0$

* $c(b_j)=[\int_0^{\pi}\exp\left(b_j*\cos\theta\right)d\theta]^{-1}$

The parameter of concentration (b_j) determines the spread of the component due to $\underset{\sim}{T}^{(j)}$. Each such component is bell-shaped. The higher is the value of b_j, the taller and slimmer is the bell. As $b_j\rightarrow 0$, the bell goes to uniform distribution over $[0,\pi]$. As $b_j\rightarrow\infty$, the bell approaches a spike at $\theta_j=0$.

The goal of including the parameter b_j is to be able to express the varying specificity of terms. Terms which are highly specific should have higher values of b_j. For example, compound terms (terms that consist of more than one word) tend to be more specific than single-word terms. [Streeter 88a] [Streeter 88b]

It is interesting to note that if we set $b_j=1$ for $j=1,...,k_o$ and use uniform prior $\prod_o$, then f_1 makes the relevance density method equivalent to a particular instance of averaging. The documents are ranked in order of their cosine with a single query vector which is the sum of term vectors normalized by their lengths.

If $b_j=b$ then in the limit as $b \rightarrow \infty$, both f_1 and f_2 rank documents in the same order as the Nearest Neighbor rule. That means, that if we have two documents whose vectors are $\underset{\sim}{D}$ and $\underset{\sim}{D}'$ respectively, then $\underset{\sim}{D}$ is ranked ahead of $\underset{\sim}{D}'$ whenever the distance (measured by cosine) between $\underset{\sim}{D}$ and the $\underset{\sim}{T}^{(j)}$ nearest to $\underset{\sim}{D}$ is smaller than the distance bethween $\underset{\sim}{D}'$ and the $\underset{\sim}{T}^{(j)}$ nearest to $\underset{\sim}{D}'$. If $\underset{\sim}{D}$ and $\underset{\sim}{D}'$ are equidistant from their respective nearest neighbors among the query terms $\underset{\sim}{T}^{(j)}$ $j=1,..,k$, then the ranking is determined by the distances to the respective second nearest neighbors etc.

The Nearest Neighbor rule above tells us that if the terms of the query are treated equally and the parameter of concentration goes up, the density does become multimodal. Ultimately, there is a separate mode over each of the term vectors.

II.3 Number of operations required by Vector Averaging and Relevance Density Method.

It is interesting to compare vector averaging and relevance density methods on the basis of the number of operations required in order to carry out a single "stage" of the retrieval (e.g. processing the original query or relevance feedback). Let us assume that there are H documents and K terms in the collection. Also, we assume that the document-term space has n dimensions and that there are k terms in the query (or k relevant documents marked relevant by the user).

II.3.1 Vector Averaging.

For vector averaging, assuming the dot product is used as the measure of similarity, the total number of operations involved is 3(k+H)n plus a single sort of H objects. These include (k+H)n multiplications, (k+H)n additions and (k+H)n retrievals.

If cosine is used as the measure of similarity, instead of dot product, the number of operations is higher, since the magnitude of the vectors must be calculated and the dot product divided by the magnitudes. H retrievals and H divisions would be added to the total number of operations.

II.3.2 Relevance Density Method

We will assume that the matrix of term-document cosines and the matrix of document-document cosines have been calculated in advance and stored. (This is reasonable, since these matrices are going to be used repeatedly and their storage space is a small fraction of the storage space occupied, say, by the texts of documents stored in the system). The exact number of operations to rank the documents depends on whether we are processing the original query or relevance feedback. If we are processing the original query and $\prod_o \neq$uniform, we must perform (k+1)H retrievals, (k+1)H multiplications, kH additions, and either kH additions and kH operations of taking a number to a power or kH multiplications and kH operations of exponentiation (Whether we perform one or the other of the two options depends on whether we are using f_1 or f_2). Finally, a sort of H objects must be performed. In total, there are (5k+2)H operations plus the sort. Processing the original query when $\prod_o$=uniform takes H fewer retrievals and H fewer multiplications than calculated above and processing relevance feedback takes H fewer retrievals than the above calculation. Thus, relevance density requires no more than (5k+2)H operations, of which no more than (k+1)H are retrievals.

We observe, that vector averaging required on the order of 3(k+H)n operations, including (k+H)n retrievals. As a rule, the number of dimensions required for adequate vector representations of the terms and documents of a large collection is much larger than the number of terms in a query or the number of documents the user marks relevant at a given stage of retrieval. In our notation, this means that n is much larger than k. Since H, the number of documents can also be very large, this means that vector averaging can require many more operations per stage than the relevance density method. While the relevance density has some computationally intensive operations (e.g. exponentiation), vector averaging has more retrieval operations, which tend to be time consuming.

III. Testing Results.

The most important question about any method of information retrieval is: how well does it perform in practice? In order to answer this question fully, the method proposed in this work should be tested on numerous large document collections, with queries actually constructed by various users. Running the full ensemble of tests is beyond the scope of this paper. To date we have conducted some simple tests comparing the relevance density method to the vector averaging method implemented at Bellcore on two information retrieval collections. Terms and documents in each collection were represented by 100-dimensional vectors in a space constructed by the Singular Value Decomposition method. For each collection we chose the vector similarity measure and the weighting of the terms (in vector averaging), which have performed best on the previous tests of vector averaging on that collection.

III.1 Tests on the ADVISOR collection

ADVISOR is an expert locating system developed at Bellcore [Streeter 88a] [Streeter 88b]. The system responds to a query by identifying departments within Bellcore, a large diversified research and development company, which are best suited to respond to the query. 104 of the company's departments were represented by annual project write-ups and abstracts of their technical papers. The collection contained 728 documents indexed by 7,100 terms. New abstracts were collected and used as test queries. There were 263 such queries. The measure of performance for each test query was the rank which the system assigned to the first retrieved document produced by the same department that the query came from. Thus, if the method were perfect, the correct department's rank would be 1; by chance the rank would be 52.

Using the formulae developed in the preceeding section, we can calculate the average number of operations per stage of processing required by the relevance density and the vector averaging ranking methods. In the queries used with this collection, the mean number of terms per query was 48.9 (approx. 49). So, using $k=49$, $n=100$, $H=728$ we get:
number of operations per stage for vector averaging $= 3(k+H)n =$
$= 3\cdot(49 + 728)\cdot 100 = 233{,}100$. Of these, $(k + H)n=77{,}700$ are retrieval operations.
number of operations per stage for relevance density $\leq (5k+2)H =$
$= (5\cdot 49 + 2)\cdot 728 = 179{,}816$. Of these, $(k+1)H = 50\cdot 728 = 36{,}400$ are retrieval operations. The total number of retrievals are both smaller for relevance density method.

For each query the documents in the collection were ranked separately by the two methods. The relevance density method was used with a uniform (constant) prior, and sampling density function $f_2(Q_o|D)=\sum_{j=1}^{k_o} \exp(b_j\cdot\cos(\angle(\underset{\sim}{D},\underset{\sim}{T}^{(j)})))$. Equal weights were used on all the terms.

When we used the relevance density method with the same parameter of concentration, b=1, on all the terms, Vector averaging and the relevance density method produced similar results. Both methods predicted the correct department with median rank 3. The 75th percentile for the rank of the correct department was 19 for vector averaging and 17 for relevance density method. But when we used the relevance density method with b=1 for single terms (terms which consist of a single word), and b=2 for compound terms (terms which consist of more than one word, e.g. *artificial intelligence*), the median rank was reduced to 2 and the 75th percentile to 9.

The specific task for which the ADVISOR was designed is not likely to exploit fully the possibility of multimodal relevance distribution, since it is not necessary to identify all the documents relevant to the query. The task is successfully accomplished if only one of the relevant documents is given a high rank. Thus, it is especially encouraging that the relevance density method performs so well in this context.

III.2 Tests on the TIME collection.

The second test was the usual task of retrieving documents in response to a query in one of the standard document sets (TIME). TIME is a collection of articles on international affairs from 1963 TIME magazine (a weekly news magazine in the United States). The collection includes 425 documents and is indexed by 10,337 terms. The collection came with 83 queries and relevance judgements, which we used for testing.

Once again, we can use the formulae developed in the previous section to compute the average number of operations per stage of processing required for the relevance density and the vector averaging ranking methods. In the TIME collection queries the average number of terms per query was 7.9 (approx. 8). So,

using $k=8$, $n=100$, $H=425$ we get:
number of operations per stage for vector averaging $= 3(k+H)n =$
$= 3(8 + 425)\cdot 100 = 129{,}900$. Of these, $(k+H)n=43{,}300$ are retrieval operations.
number of operations per stage for relevance density $\leq (5k+2)H =$
$= (5\cdot 8 + 2)\cdot 425 = 17{,}850$. Of these, $(k+1)H = 8\cdot 425 = 3{,}825$ are retrieval operations. The total number of operations and the number of retrievals are an order of magnitude smaller for the relevance density method as compared to the vector averaging method.

In response to each query, the documents were ranked separately by the vector averaging method and by the relevance density method. The relevance density method was used with two different sampling density functions:
$f_1\left(\underset{\sim}{Q}|\underset{\sim}{D}\right) = \sum_{j=1}^{k} \cos\left(\angle\left(\underset{\sim}{D},\underset{\sim}{T}^{(j)}\right)+1\right)^{b_j}$ and $f_2\left(\underset{\sim}{Q}|\underset{\sim}{D}\right) = \sum_{j=1}^{k} \exp\left(b_j * \cos\left(\angle \underset{\sim}{D},\underset{\sim}{T}^{(j)}\right)\right)$.
We tried different values of the parameter of concentration b, but in all cases the same value of b was used on all the terms of all queries and the same value was used on all the relevant documents in the feedback step. All the components had the same weights. A uniform (constant) prior was used in all cases.

In the tests on the TIME collection we measured performance after the collection was ranked in response to the original query and after a single feedback step. The feedback step consisted of comparing the documents ranked in top 15 to the list of documents known to be relevant to the query. If this group of 15 documents contained some documents relevant to the query, a new ranking for the entire collection was computed.

The measure of success for each query was precision averaged over 3 levels of recall (25%, 50%, 75%). Note: *precision* is the proportion of the documents retrieved that are relevant to the query. *Recall* is the percentage of the total number of documents in the collection relevant to the query, that is included in the group of documents retrieved in response to the query.

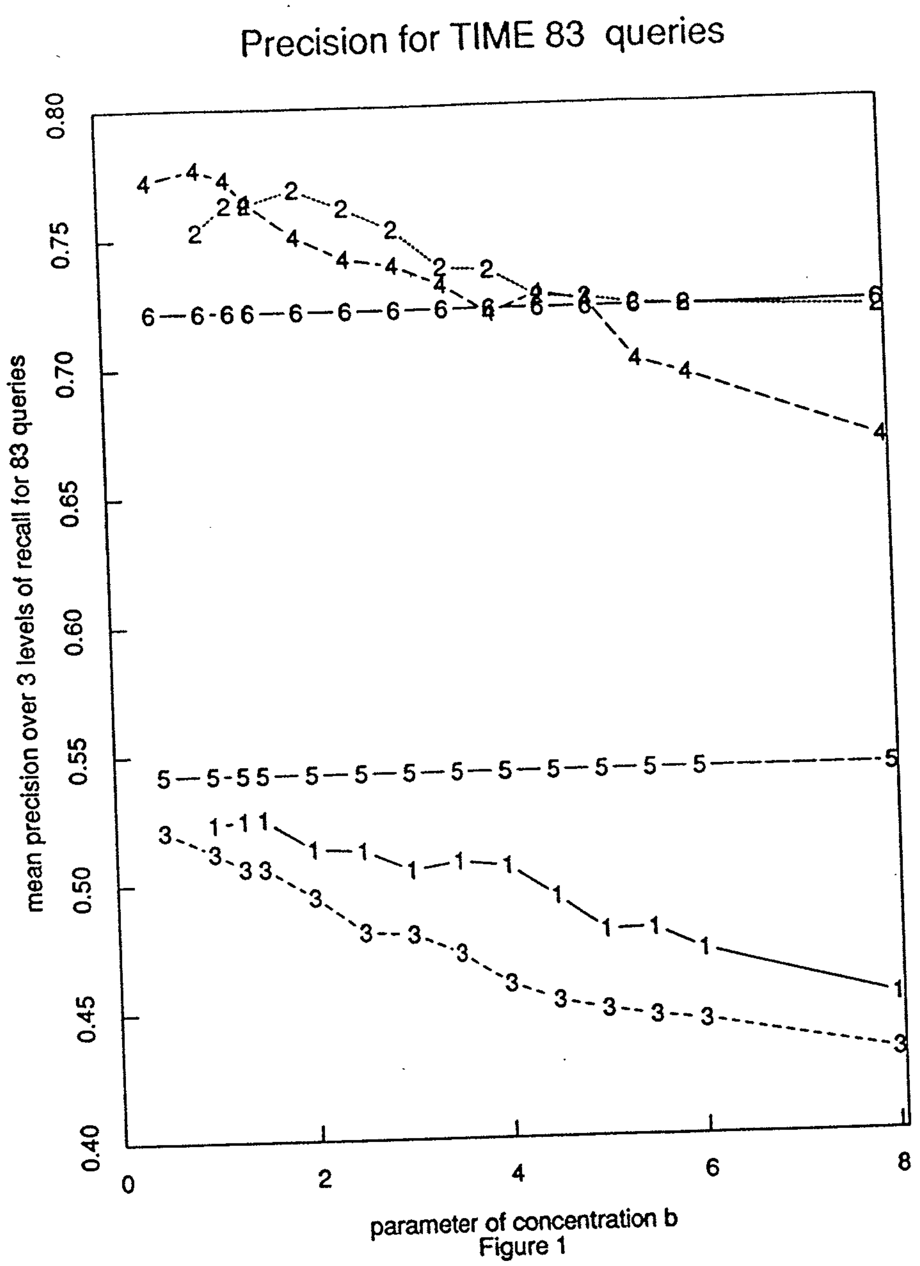


Figure 1

Curve 1: relevance density, $f_1=\sum_{j=1}^{k}(\cos(\theta_j)+1)^b$ used as sampling density, original query only.
Curve 2: relevance density, $f_1=\sum_{j=1}^{k}(\theta_j+1)^b$ used as sampling density, original query and one feedback step.

Curve 3: relevance density, $f_2=\sum_{j=1}^{k}\exp(b\cdot\cos\theta_j)$ used as sampling density, original query only.
Curve 4: relevance density, $f_2=\sum_{j=1}^{k}\exp(b\cdot\cos\theta_j)$ used as sampling density, original query and one feedback step.

Curve 5: vector averaging, original query only.
Curve 6: vector averaging, original query and one feedback step.

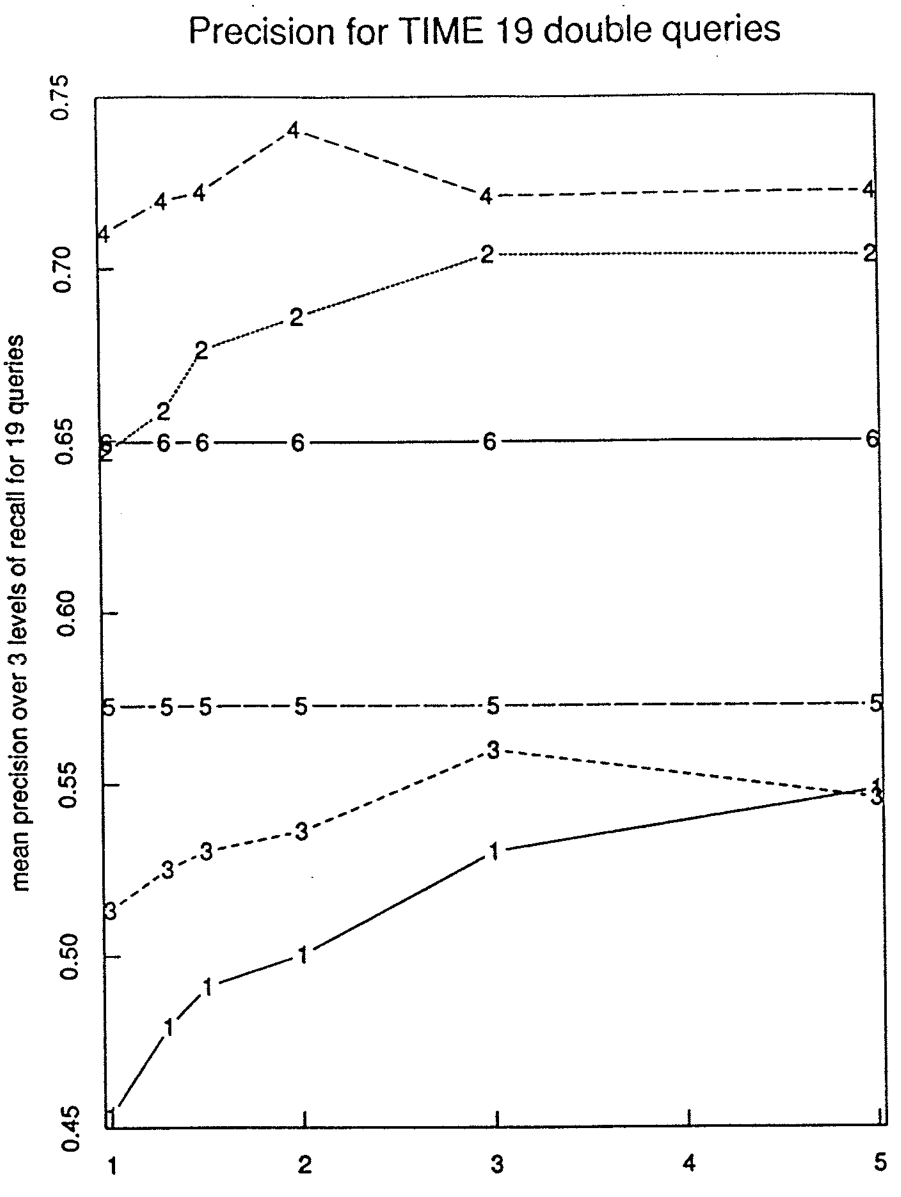


Figure 2

Curve 1: relevance density, $f_1=\sum_{j=1}^{k}(\cos(\theta_j)+1)^b$ used as sampling density, original query only.
Curve 2: relevance density, $f_1=\sum_{j=1}^{k}(\theta_j+1)^b$ used as sampling density, original query and one feedback step.

Curve 3: relevance density, $f_2=\sum_{j=1}^{k}\exp(b\cdot\cos\theta_j)$ used as sampling density, original query only.
Curve 4: relevance density, $f_2=\sum_{j=1}^{k}\exp(b\cdot\cos\theta_j)$ used as sampling density, original query and one feedback step.

Curve 5: vector averaging, original query only.
Curve 6: vector averaging, original query and one feedback step.

Precision vs. recall curves are the most commonly used indicators of performance in Information Retrieval. We calculated mean values of the 3-point precision average across all the queries to give us an overall summary of performance of each run. The results are summarised in Figure 1.

Of the 83 queries from the TIME collection, 23 have only one relevant document. Precision averaged over three levels of recall is not a good measure of performance for such queries, since there can only be one meaningful level of recall, i.e. 100%. Thus, we measured performance on the 60 queries which had two or more relevant documents using b=1.5 in the relevance density function. The results agree with those obtained when all 83 queries are used.

In order to observe the behaviour of the ranking methods in a situation where the query is known to have relevant documents in more than one area of the document-term space, we constructed "double queries" by joining together the texts of pairs of queries and joining together the corresponding lists of relevant documents. From the resulting double queries we selected 19 double queries whose two component queries had their respective query vectors (as computed by the vector averaging method) roughly orthogonal to each other, i.e. the cosine of the angle between the two query vectors was between -0.1 and 0.1. The results are summarised in Figure 2.

In both tests on the TIME collection, vector averaging performed better when only the original query was used in retrieval. We suspect that using the same concentration parameter and the same weight on all the terms of the query is inappropriate in the relevance density method. Query terms were weighted by entropy weights in the case of vector averaging. relevance density method performed better if the feedback step was included and b set at about 2.0 for f_1 and at about 1.5 for f_2. Overall, f_2 with b close to 1.5 gave better performance than f_1. While the differences in performance among the different methods were not very dramatic, the trends are consistent across the values of b used in the sampling desnities of the relevance density method and between the two tests run on the collection.

In a large collection we would expect vector averaging to perform very poorly in the case of a double query. We suspect that the good performance of vector averaging on the double queries in the TIME collection might be due to the fact that the TIME collection has relatively few documents compared to the dimensionality chosen for the document-term space, thus the increase in the number of relevant documents when the queries were doubled resulted in a big enough improvement in precision average to offset the effect of averaging two disparate sets of query terms.

When b=1, using f_1 as the sampling density amounts to using a particular form of vector averaging. Thus, in this case we expect the relevance density method to give results very similar to those obtained by the vector averaging method. This is the case in the test using 83 queries with and without feedback and in the test using the 19 double queries with feedback. Unexpectedly, f_1 with b=1 gives a noticeably poorer performance when the 19 double queries are used without the feedback step. Again, this might be due to inappropriate weighting of the query terms.

VI. Summary

The preliminary tests of the relevance density method of ranking documents are encouraging. The f_2 sampling function with the value of the parameter of conentration b set between 1 and 2 gave the best results. We expect further improvements if unequal values of b and unequal weights on terms are used and if prior information on the users who generate the queries is incorporated into the relevance density.

We do not know whether the two document collections and the test queries we have used are representative of the "field conditions" i.e. a real library and queries generated by its population of users. In particular, we do not know how often a query has to be answered by documents whose vectors occupy several separate areas in the document-term space. However, there is no reason for the relevance density method with a well chosen sampling density function to do worse than vector averaging in the cases where the vectors of the relevant documents do all reside in a single area. On the other hand, the proposed method makes more sense when averaging is not appropriate.

REFERENCES

[Bookstein 82] Bookstein A. "Explanation and Generalization of Vector Models in Information Retrieval" *Research and Development in Information Retrieval.* Proceedings Salton G. and Schneider H.J. (eds.) May 1982 pp. 118-132.

[Borodin 68] Borodin A., Kerr L. and Lewis F. "Query Splitting in Relevance Feedback Systems. *Scientific Report No. ISR-14*, Section XII, Department of Computer Science, Cornell University, October 1968

[Chow 82] Chow D. and Yu C. T. "On the Construction of Feedback Queries" *Journal of the Association for Computing Machinery* 29 (1), January 1982 , pp. 127-151.

[Furnas 88] Furnas G.W., Deerwester S., Dumais S. Landauer T.K., Harshman R. A., Streeter L.A. and Lochbaum K.E. "Information Retrieval Using a Singular Value Decomposition Model of Latent Semantic Structure." Proceedings of SIGIR '88, Grenoble, France, pp.465-480.

[Jones 87] Jones W. P. and Furnas G. W. "Pictures of Relevance: a Geometric Analysis of Similarity Measures" *Journal of the American Society for Information Science*, 38(6), November 1987, pp. 420-442.

[Kane-Esrig 90] Kane-Esrig Y. , "Modeling and Using Information About the Height of a Probability Density over Selected Points of its Support with a Motivating Problem from Information Retrieval", *Ph. D. Thesis,* Field of Statistics, Cornell University (In preparation).

[Losee 87] Losee R. M. "Probabilistic Retrieval and Coordination Level Matching" *Journal of the American Society for Information Science* 38(4), July 1987, pp. 239-244.

[Noreault 81] Noreault T., McGill M. J. and Koll M.B. "A Performance Evaluation of Similarity Measures, Document Weighting Schemes and Representations in a Boolean Environment", *Information Retrieval Research,* Butterworths and Co, London, 1981.

[Salton 68] Salton G. *Automatic Infomation Organization and Retrieval* McGraw Hill, New York, 1968.

[Salton 83] Salton G. and McGill M. J. *Introduction to Modern Information Retrieval* McGraw Hill, New York, 1983.

[Streeter 88a] Streeter L. A. and Lochbaum K. E. "An Expert/expert-locating System Based on Automatic Representation of Semantic Structure" *Proceedings of the Fourth Conference on Artificial Intelligence Applications*, San Diego Ca., March 14-18, 1988.

[Streeter 88b] Streeter L.A. and Lochbaum K.E. "Who knows: A system based on automatic representation of semantic structure". *RIAO* 88: *User-Oriented Content-Based Text and Image Handling*; Massachusetts Institute of Technology, Cambridge MA, March 21-24, 1988, pp. 379-388.

[Van Rijsbergen 79] C. J. Van Rijsbergen *Information Retrieval* (2nd *ed.*), Butterworths, London, 1979.

ISSN 0105-8517

Proceedings of the 12th IRIS – Part II

*I*nformation systems *R*esearch seminar *I*n *S*candinavia

August 13-16, 1989, Skagen, Denmark

Susanne Bødker, ed.

DAIMI PB – 296-II
December 1989

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

Contents

PART II

ON THE CREATIVITY SUPPORT PROVIDED BY CASE TOOL ENVIRONMENTS

Pentti Kerola, Pasi Kuvaja and Ari Vaulo
Department of Information Processing Science
University of Oulu, Linnanmaa
SF-90570 OULU, Finland

Abstract

The fundamental question asked in this paper is "What could the creativity concept mean in the IS design process and do CASE tools support or prevent it?" The conceptual content of creativity is based on the Heidegger-Winograd-Flores socio-linguistic foundations of human effort and design activity complemented with Kolb's polylectic features of experiential learning. As an experimental portion of the research, the Deft CASE tool is utilized for the specification process of the IFIP Inventory Control and Purchasing System Case. Three researchers implement a 'small-scale' specification effort, in which each has his own individual approach, but the research reflections are made together.

This results in a definition of the concept of creativity on the three levels: design task, individual and team. The paper emphasizes that creativity exists or not in terms of rapports between an individual and her/his activity environment; i.e. design tasks, CASE tools and other human participants. Because of different human experiential styles and levels at which creativity may be applied, the total support/prevention impact of CASE tools shows great situational variety, including contradictory features.

1 INTRODUCTION

It is claimed that the CASE (Computer-Aided Software Engineering) tools and systems ease, support and integrate the analysis, design, generation and maintenance of computer-based information systems. Both the general literature (e.g. Fisher 1988, McClure 1988, Huling 1987 and Chikofsky & Rubenstein 1988) and research reports (e.g. Bubenko 1988, Case 1987, Kumar & Welke 1988) describe the technical features of CASE tools and their connection with existing systems design methodologies or try to understand their functional features through different frameworks or levels of abstraction. Thus the general approach in recent papers has either been commercial or reflects the ordinary technical and functional approach to CASE tools. Only Windsor (Windsor 1986) considers systems engineering tools from the viewpoint of systems analysts or designers.[1]

While CASE tools can offer significant benefits, we do not yet possess a panacea for systems development and use. Many contradictory problems have also arisen, as is typical when computer support and at least partial automation are planned for human activities. Shall we support, prevent or even kill the creativity of system analysts and designers? Are CASE systems changing work structures away from their emphasis on craftmanship and towards industry, and what does it mean for the creativity of people participating in this area of applications for engineering efforts of the CAD/CAM type? These questions form the general basis and main motives for this research.

This paper includes research problem definition and general description of the experimental small-scale study performed in section 2. Then in section 3 the conceptual framework as a research instrument for the experimentation will be defined and discussed in relation to each other. Experimentations with reflective observations will be presented in sections 4. Final conclusions and research findings will then be presented in the last section.

2 PROBLEM AREA AND EXPERIMENTAL RESEARCH SETTING

The meaning of creativity varies greatly between different people – and especially among IS professionals! This is therefore a challenging and difficult topic of research. In this study we are constrained to the early specification subprocesses of IS design, which are of special interest to us because of the sensitivity of their relations to creativity. Within this research object area our purpose is to study and discuss the concept of creativity in the context of CASE tools utilization. Our problem may be formulated as a question: "What could the creativity concept mean in the IS design process, and do CASE tools support or prevent it?

[1]He defines seven advantages and five disadvantages of automated tools for systems design.

Since there is no theoretical agreement about the concept of creativity and practically no earlier research efforts exist, we decided to use experimental approach with a conceptul framework as an analyzing instrument for the development process, and MBTI self-assesment instrument (Myers and Briggs 1962) for measuring our personal learning styles during the experiments. Our conceptual preunderstanding of creativity as a specified feature of human effort is based on two major references: Winograd-Flores *"Understanding Computers and Cognition – a new foundation for design"* (1986) and Kolb *"Experiential Learning – experience as the source of learning and development"* (1984).

Our approach follows the typical hermeneutic circle, where understanding of the conceptual framework, object system under development and development tool will increase and result higher understanding of the problem area defined. Experiments with reflective observations are based on the case of the IFIP Inventory Control and Purchasing System[2] (Olle et al 1988) and the DEFT Environment[3] (1988) with Mac implementation. Each participant will implement his own individual specification process, with personal reflections and notes to be added at team meetings.

3 CONCEPTUAL FRAMEWORK AND MEASUREMENT PARAMETERS

3.1 Introduction

In order to investigate creativity in IS development utilizing a CASE tool environment we take as our starting point Mathiassen's (Mathiassen 1988) synthesis of the creativity in IS development, that is originally based on Fairley's (Fairley 1985) ideas. In these apprroaches two different elements in IS design have been identified. First of these is **creative element** that comprehends conceiving and planning out of mind, while the second is **describing element** including making a drawing, pattern or sketch of. Both of these are argued to form mutual dependent and inseparable elements of the design work. The creative element is also claimed to be so dependent on the designer's personality and previous experience that there are limited possibilities for strengthening it through application of systematic methods, which in our approach mean systems development methodologies included and supported by Deft CASE tool

[2]A small company maintains a stock of various items for its own use. For the purpose of this illustration, all stock is kept at one location. As and when necessary, the company orders new supplies from one of its suppliers. It therefore keeps track of the kinds of supplies available for each supplier and it also keeps track of its current stock on hand for each kind of item it needs.

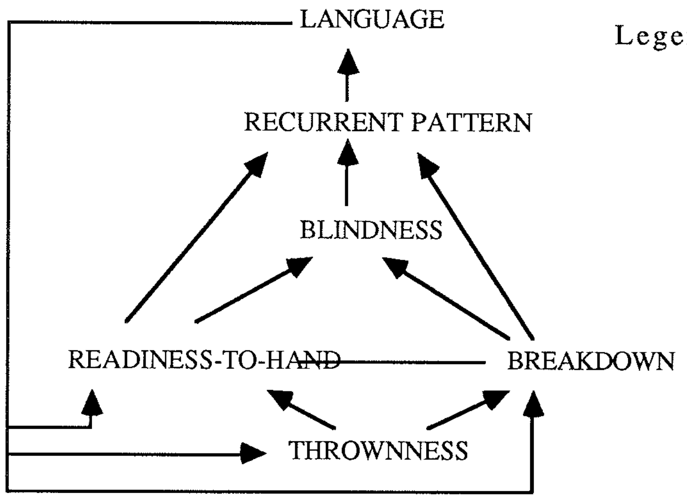
[3]Deft is a CASE tool set which offers a set of integrated tools available for online application analysis and design. Deft automates the documentation and diagramming process, and designers can produce the system documentation, data flow diagrams, data base design, program structure definitions, forms and screen design in quite a short time. It is possible to receive proof of correctness at every stage in the development process.

environment. Here we will try to analyze IS design process supported by the CASE tool environment by applying Heideggerian concepts of human effort in order to find out that does the CASE tool environment support or prevent the creativity. This means actually a partial analysis of the mutual dependency between the creative element and describing element, when the creative element in our experiments is the design of Inventory Control and Purchasing system using the IFIP case descriptions and our former experiences as the source of information, and when the descriptive element is the use of Deft CASE tool environment. Designer's personality, that is also one of the key elements as decsribed above, we will investigate by using Kolb's experiential learning style analysis implemented as a self-assesment by MBTI indicator (Myers et. al. 1962).[4][5] As a summary of these efforts we will try to find out the limits of the possibilities to strengthen the creative element through application of CASE tool environment in ISD.

From this starting point we will present here Heideggerian concepts of human thought, Kolb's experiential learning style model and their relationships in a polylectical framework. Then, as a final summary of our preunderstanding of the conceptual framework, we will describe the relationship between the creative element and creativity by using a role-based description.

3.2 Fundamental Heideggerian concepts of human effort

For our experimental purposes the following basic concepts (Figure 1) are selected for definition and characterization:



Legend: $a \longrightarrow b$

concept a (and its precedents) has an essential impact on the definition of concept b (Note the circularity !)

Figure 1

[4]The MBTI results were transformed to the Kolbian styles in the following way:
- concrete experience emphasizes perception and feeling,
- reflective observation emphasizes introversion and intuition,
- abstract conceptualization emphasizes thinking and judgement, and
- active experimentation emphasizes extraversion and sensing.

[5]See also Figure 2.

Our intention here is to pick out only those aspects relevant to our examination of creativity in the design process.

Thrownness (Winograd-Flores pp. 33-35, 71, 97) is the essence of the human being's prereflective experience of being thrown into a situation in which he is expected to act. We are always engaged in acting within a situation, without the opportunity to fully disengage ourselves and function as detached observers. Even what we call disengagement occurs within thrownness. We can't escape our thrownness, but can only shift our domain of concern. Our acts always take place within thrownness, and cannot be understood as the results of a process (conscious or unconscious) of representing, planning and reasoning.

Readiness-to-hand (pp. 36-37, 71-73) is our ability to utilize (earlier) experiences preconsciously as they involve present at hand objects and their properties, which are currently to hand.

Breakdown equals to **unreadiness-to-hand** (pp. 77-79) is a moment of interruption in our habitual standard comfortable state of being in the world that triggers our reflections. Breakdowns serve an extremely important cognitive function, revealing to us the nature of our practices and equipment. New design are created and implemented only in the spaces that emerge in the **recurrent structure of breakdowns**. A design constitutes an interpretation of a breakdown and a committed attempt to anticipate future breakdowns. Most important is the fundamental role of breakdowns in creating the space for what can be said, and the role of language in creating our world – recognizing the fundamental importance of the shift from an individual-centred concept of understanding to one that is socially based.

Blindness (pp. 97-100) is the side-effect of our reflections, attributable to the fact that our subjective view is always limited to what can be expressed in the terms and language we have adopted – it is necessary and inescapable, but we seldom are explicitly aware of it. Reflective thought is impossible without the kind of abstraction that produces blindness. We can never have a full explicit awareness of our prejudices, however.

The basic function of **language** (pp. 50, 68-69) as a system of orienting behaviour is not the transmission of information or the description of an independent universe about which we can talk, but the creation of a consensual domain of behaviour between linguistically interacting people. No thing exists except through language. In saying that a 'thing' exists, we have already brought it into the domain of articulated objects and qualities that exists in language and through the structure of language, which is constrained by our potential for action in the world.

The distinctions made by language are not determined by any objective classification of situations in the world, but arise from repetitive **recurrent patterns** of breakdown in human effort. Words arise to help anticipate and cope with these breakdowns. E.g. Finnish reindeer herders have a large number of distinct terms for reindeer (nine, to be exact). This is not just because they see a lot of reindeer, but because there are recurrent activities with spaces of potential breakdowns for which the distinctions are relevant.

3.3 Experiential styles and their polylectics

The major source of pattern and coherence in Kolb's framework is its notion of the multidialectical (polylectic) nature of the human evolutionary process. Dialectic here means structure and interaction between two components which are at the same time contradictory and complementary (concrete experience versus abstract conceptualization, reflective observation versus active experimentation) (See Figure 2).

With time, individuals develop their typical possibility-processing structures (recurrent patterns) in such a way that the dialectic tensions are consistently resolved in a characteristic fashion. As a result of their genetics, past life experiences and social environments, people develop experiential learning and human information processing styles that emphasize certain abilities and characteristics more than others. In this way Kolb defines the four basic categories of experiential styles:

> "The convergent learning style relies primarily on the dominant abilities of abstract conceptualization and active experimentation. The greatest strength of this approach lies in <u>problem solving, decision making</u> and the practical application of ideas. We have called this learning style the <u>converger</u> because a person with this style seems to be best in situations like conventional intelligence tests, where there is a single correct answer or solution to a question or problem. In this learning style, knowledge is organized in such a way that through hypothetical-deductive reasoning it can be focused on specific problems.
>
> The divergent learning style has the opposite learning strengths, emphasizing concrete experience and reflective observation. The greatest advantage of this orientation lies in imaginative ability and <u>awareness of meaning and values</u>. The primary adaptive ability of divergence is to view concrete situations from many perspectives and to organize a multitude of relationships into a meaningful "gestalt". The emphasis in this orientation is on adaptation by observation rather than action. This style is called the <u>diverger</u> because a person of this type performs better in situations that call for the generation of alternative ideas and implications, such as a "brainstorming" idea session. Those oriented toward divergence are interested in people and tend to be imaginative and feeling-oriented.

The dominant learning abilities in the assimilator style are abstract conceptualization and reflective observation. The greatest strength of this orientation lies in inductive reasoning and the ability to create theoretical models, forming assimilative disparate observations into an integrated explanation. As in convergence, this orientation is less focused on people and more concerned with ideas and abstract concepts. Ideas, however, are judged less by their practical value. It is more important that the theory should be logically sound and precise.

The accomodative learning style has the opposite strengths from assimilation, emphasizing concrete experience and active experimentation. The greatest strength of this orientation lies in doing things, carrying out plans and tasks and involving oneself in new experiences. The adaptive emphasis of this orientation is on opportunity seeking, risk taking, and action. This style is called the accomodator because it is best suited to situations where one must adapt oneself to changing immediate circumstances. In situations where the theory or plans do not fit the facts, those with an accomodative style will most likely discard the plan or theory. People with an accomodative orientation tend to solve problems in an intuitive trial and error manner, relying heavily on other people for information rather than on their own analytic ability."

In the following the Heideggerian kernel concepts are hypothesized as being related to the basic concepts of Kolb's experiential learning theory (Figure 2).

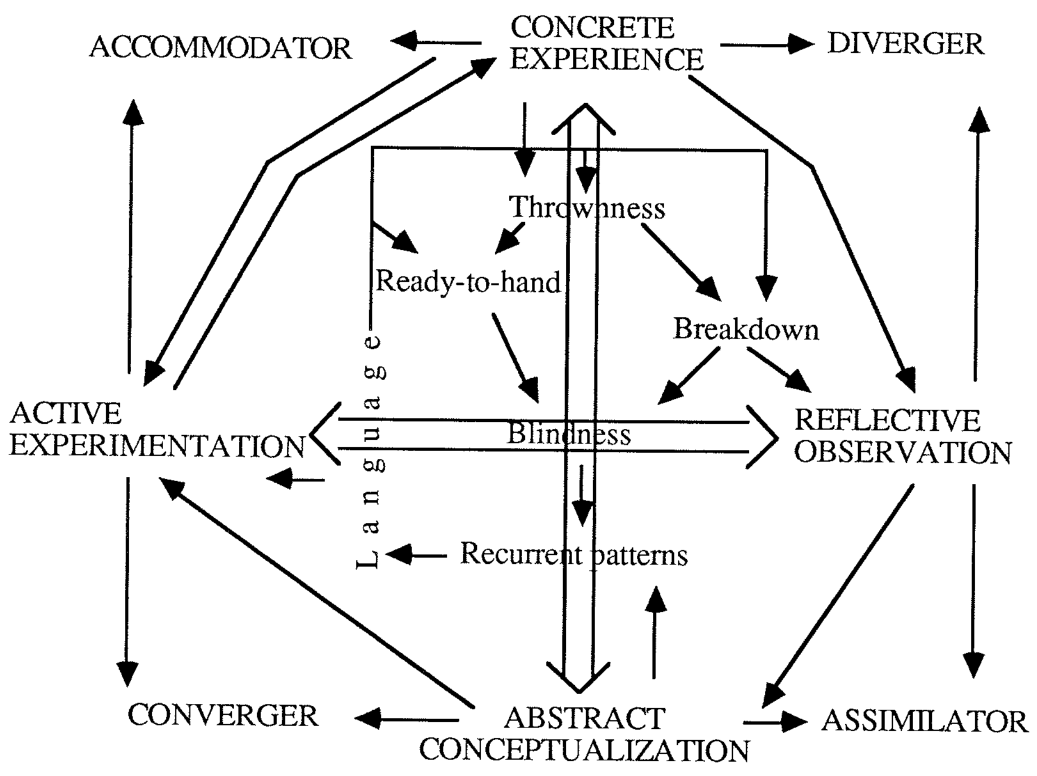


Figure 2

In the figure 2 the concept structure of figure 1 has been situated in the Kolbian two-dimensional field of tension spanned by the 'thick double arrows'. The detailed reasoning has been omitted because of the lack of space in this paper, but the definitions of the kernel concepts give enough indications of the conceptual relationships and locations. Observe in particular the location of 'blindness' in the centre of the figure, where it can be applied in any direction in the field.

Kolb's basic style categories also form dialectic pairs: accomodator versus assimilator and diverger versus converger. The elements of the pairs can be shortly described in terms of the Heideggerian concepts as follows:

- the diverger is a 'breakdown specialist'
- the assimilator is a 'breakdown analyst and pattern developer'
- the converger is a 'recurrent pattern and language user'
- the accomodator is a 'language user preferring readiness to hand' (doesn't like breakdowns !)

We have to conceive of basic learning styles as generic adaptive competences, i.e. higher-level learning heuristics that facilitate the development of a class of more specific skills required for effective performance in different human tasks and activities. **Concerning creativity, the styles are context factors influencing the selection of motives, objects of interest and operations**. Different kinds of creativity probably appear in the context of different styles, e.g. the diverger is probably creative with ideas of new values, directives etc. and non-creative with ideas of new solutions, decisions, etc. But at the same time the situation is precisely the opposite where the individual creativity development is concerned! For the individual the features of non-dominant styles can always be the essential sources of personal creativity development. The problem is how one can become aware of this.

3.4 On the creative element and creativity in design

It is interesting to find that Winograd-Flores do not use the specific terms of **create** or **creativity** at all. In everyday language, however, creativity can be described as the human quality of being able to produce new or original ideas or work in any field. When it is applied to the IS design process we can distinct the following levels at which it can be implemented:

1. Role of the IS consumer (end-user) in the creative process of using the IS and generating new ideas about the UoD,
2. Role of the IS producer (designer) in the creative process of developing new ideas about changes in the existing IS,

3. Role of the CASE tool consumer (designer) in the creative process of using the CASE tool in the creative and/or descriptive subprocesses of design.

The lower level effort can always influence on the upper level processes either positively or negatively. The roles can be implemented by different people or by the same person.

In the Heideggerian sense, the concept of creativity is now understood to be based on and related to the kernel concepts described above, and especially on the concepts of blindness, recurrent patterns and language. To **increase creativity means to motivate ourselves to extend our language skills, to reduce our blindness and to evolve recurrent design patterns in a consciously directed manner in order to release human resources for idea generation**. Because of the different levels at which creativity is applied, the total support/prevention impact of CASE tools possesses great variety, including contradictory features.

4 EXPERIMENTATION

4.1 Introduction

The purpose of this experimentation is to try to observe creativity and its kernel concepts and make them more explicit in the efforts to develop real systems. The object of the experimentation was to make specifications of the IFIP Case (Olle et al. 1988) using the Deft CASE tool environment and to make reflective observations during the process. The IFIP Case gives an illustration of a typical inventory control and purchasing application in a small company. The description includes two logically separate parts, "Business analysis" and "Systems design specifications". Only the contents of the former part of the description, referred to simply as the IFIP Case,were used in the experiment.

Each experimental case will be presented here separately. At first the experimenter is described by presenting the result of human information processing style evaluation and measurement with the MBTI instrument. Second part of the results of each experiment will be reported in the form of a table with explanatory comments. This represents "condensed raw data" of the experiment that has been produced on the basis of experimenter's diary notes. The columns in the tables include the kernel and creativity concepts, and the rows are either steps or objects of interest encountered during them. Each chronological phase of the experiment forms a step in the table. Each step can include one or more objects of interests, and the steps are coded alphabetically. The table entries are given as numbers referring to the intersections of rows and columns, if any kernel concept has been reflected in any step or its object of interest. Each numbered entry is explained with a written comment. Third part of the results of each experiment includes experimenter's subjective conclusions, especially concerning his answers to the research questions stated.

4.2 Experiment A

Designer of the experiment A described himself by the MBTI self-assesment as a strong combination of accomodator and converger.

Experimental data

Kernel Concepts Steps	-Objects of Interest	Throwness	Ready-to-Hand	Breakdown	Blindness	Language	Recurrent Patterns	Creativity
Step A	IFIP case			1		1		2
	Macintosh		3					
	Deft as a program		3					
	Using Deft in this case	4		1			4	
	Yourdon DFD	5	5	1				
Step B	IFIP case	6		1				
	Deft in this case		7					
	Yourdon DFD	5	5					
	Own SE method		6		8			8
Step C	IFIP case	6		1				9
	Yourdon DFD		5					
	Using Deft with DFD		10					10
	Deft in this case		7					
Step D	IFIP case	6	7	1				9
	Deft in this case							
	Using Deft with ERD	11		11				
	Using Deft with PSD	11		11				
Step E	IFIP case	6		1				
	Deft in this case (Deft method)		7					
	ERD and PSD descriptions	12						
	Using Deft with ERD		13					13
	Using Deft with PSD		13					13
	Using Forms editor		13					13

Table 1

1. The IFIP (Olle 1988) Inventory Control and Purchasing case description is confused. I can't figure it out, partly because of the language and partly it differs too sharply from one or two-line task descriptions which I normally use. I would have been satisfied if I had got only the first page of the IFIP case.

2. The concept of creativity was limited to the process and results. It entailed subjective creativity and I thought that experience was the only way to influence the level of creativity.
3. I have used Macintosh almost the whole of my life (four years, in effect) and for a minimum of three hours a day (only one hour during a couple of days just after 11th IRIS because I got a high fever!). I have also used almost every kind of Macintosh software and I love programming with it, so I think I'm quite experienced Macintosh user and new software products didn't confuse me.
4. In spite of the remarks in point 3 above I felt a little confused because I normally use familiar examples when I acquainting myself with new software and now I had a new case which I had never seen never before and a new piece of software (DFD versions 1 to 5).
5. The Yourdon DFD method which I chose from the Deft Design toolkit is an old friend from some courses and projects with which I have been concerned. In the first two steps I felt the case description was difficult for me because I couldn't see all the information I should include in my descriptions at once. But step by step I learned more and found answers to my problems, although at the same time I found new problems to solve. For example, I can't find all information that an end user would in my opinion need from the IFIP case.
6. I felt unhappy because there was too much information in the IFIP case, so that I could not use my imagination and intuition.
7. I found new properties in Deft (text pages, how to use the dictionary, etc.) and I felt happy and able to alter earlier descriptions (DFD versions 6 to 8).
8. I used Yourdon DFD only as a description tool, because of my background as a hacker type of programmer. I usually prefer intuition in solving problems and I easily abandon all the cookbooks and methods that someone else has developed (see point 2 above).
9. In a team meeting I found great differences between the team members in their understanding and solving of problems, although I couldn't see any good reasons for these differences. As a result, I found that the team members got different results in externally the same situation and after spending quite similar amounts of time.
10. I found myself to be familiar with the Deft DFD Editor and can now work with it easily, but it's unnecessary now because I think my IFIP case DFD's are ready and I'm moving towards the next step in the software development method included in the Deft toolkit. I could see easily the next steps in using Deft for the software development process with the IFIP case style of problem description (I think it need not necessarily be this kind of description) (DFD versions 9 to 14).
11. Although I now know the whole Deft design process, there are still little difficulties with detail in using the description tools and methods. The Entity-Relationship diagrams and HIPO graphs are not so difficult or unfamiliar, but using them for first time in practice is quite a slow process and I must think what to do at almost every step when advancing in

the IFIP case description.

12. I feel a little unfamiliar with the ERDs and PSDs (Program-Structure-Diagram) because I haven't used these as much as the DFDs.
13. The ERD, PSD and Forms editors were easy to learn after quite a long period of practice with the DFD editor. I needed to make 14 versions of the DFDs, but only 4 ERDs, 3 PSDs and only 1 Form layout before I was satisfied. It is easy to be creative when you have feeling that you can control your tools.

<u>Subjective conclusions</u>

1. Role of the Information System consumer (end user) in the creativity process in using the IS?
   End users have a view of the information requirements of their work, but they cannot usually explain these needs clearly. If end users have a wide view of their work they can easily give us a large number of examples and in that way help in the systems design process.

2. Role of the Information System producer (designer) in the creativity process involved in using the IS?
   The Information System producer has all his experience and skills which he can put into the systems design process. He can also obtain hidden information from end users (information system consumers).

3. Role of the CASE tool consumer (designer) in the creativity process involved in using the IS?
   The CASE tool can support or prevent creativity. I found a three phase model in my own work with the new case tool:
   1. Learning phase. At the beginning the CASE tool, like all new methods and tools, prevent creativity because of you must pay so much attention to learning it. This can be seen in point 4 of my case experiment.
   2. Inspiration phase. In the second phase it is possible to find very high creativity because of eagerness to work with the new tool. I found this from points 10 and 13 of my case experiment.
   3. Boredom phase. When the high inspiration has gone you become tired of all these methods and tools and you want to throw them all away and forgot everything. But there is still an automatic tool which can help you, and therefore you have more power to put into the design process instead of drawing those awful diagrams manually. It is impossible to show easily where this phase was visible in my case experiment, but I have feeling that there were signs of this kind in the air.

4.3 Experiment B

Designer in the experiment B described himself by the MBTI self-assesment as a combination of accomodator and diverger.

Experimental data

Steps	Kernel Concepts and Creativity - Objects of Interest	Thrownness	Ready-to-Hand	Breakdown	Blindness	Language	Recurrent Patterns	Creativity
Step A	IFIP case			1				
	Introduction to Deft			2				
	Wall graph technique		3			3		
	Gane-Sarson methodology		4	4		4		
Step B	Gane-Sarson methodology	5		5			5	
Step C	IFIP case description	6		6			6	
	First trial with Deft			7				
	ER methodology			8				
Step D	Creativity components			9				9
Step E	IFIP case / Chapter 3		10				10	
Step F	Gane-Sarson methodology		11				11	
	First DFD with Deft	11				11		
Step G	Six DFD versions with Deft	12	12				12	
	IFIP case / Chapter 3						13	
	Reflections				14			

Table 2

1. The IFIP Case describes the application explicitly but not clearly as a part of its environment, because the environment is described only implicitly. The purpose of the application is also defined implicitly as being to support a company in ordering new supplies as necessary from one of its suppliers. When the company makes a decision to request new supplies, this is based on information produced by keeping track of the kinds of supplies

available from each supplier, and the current stock of each kind of item it needs. Thus, in order to support this decision process the application under design should produce reasonable information to allow at least the following kinds of decision to be made:

1. When are new supplies needed and when should they be ordered ?
2. What items are needed and should be included in the new supplies ?
3. Who are the suppliers who are able to supply these items ?

When I had to read nearly all of this information between the lines, I realized that the IFIP Case included hidden (or implicit) information, like a real systems design situation. I therefore realized that I needed a new way of analyzing them more carefully in order to be able to continue my design work.

2. Because the initial purpose was to make experiments with the Deft case tool environment, I tried at first to use it for a deeper analysis of the IFIP Case description, starting with chapters 1 and 2. By doing so, I hoped to get an answer to whether it was possible to analyze and describe an application as a part of its task and/or user environment with this computer-aided software engineering environment. This was in practice my first attempt with the design methodologies and description techniques included in Deft, and I found that the most important thing was to be able to use the methodologies so that the design process could go on and I could continue to make the descriptions. Thus, I started a learning process for becoming acquainted with them. At the time I was also looking at another design and/or description technique that was already ready to hand.
3. I took a step back with Deft and turned to using the wall graph technique that was already known to me on both the conceptual and the practical level. I made two wall graphs describing the explicit contents of chapters 1 and 2 of the IFIP Case description. These graphs strengthened the need for more detailed analysis of the IFIP Case in order to find out its hidden information contents.
4. In parallel with point 3 I started to learn the Deft software and recognized that I knew all of the methodologies included in it, but only on the conceptual or theoretical level, so that I had no actual prior practical experience of their use for systems design. Based on the literature available on both the Yourdon (Yourdon 1979) and the Gane-Sarson (Gane and Sarson 1979) methodologies, I chose the latter as my design and description methodology concerning potentially both the functional analysis of the object system, and the activity analysis and design of the application itself. This choice was based on the level of the detailed description of how to use the methodology in practice, as I was more satisfied with the material describing the Gane-Sarson methodology. At the time I also found that Deft does not really include these methodologies or explicitly support their use, but contains and supports only the use of the description languages included in them. I also recognized that I must know the methodology very well before I can draw descriptions in the right way.

5. When learning the Gane-Sarson methodology I systematically trained myself for the use of Deft with all its methods for systems design. I made a review of the literature on the Gane-Sarson methodology and learned it. This learning period led me to realize that potentially the same treatments would be needed for all the other methodologies included in Deft if I wished to use them later on.
6. Since, I felt that I still did not have a thorough understanding of the application to be designed as a part of its environment, I decided to find out and use the literature that describes general principles of materials handling and information processing in a typical organization. This meant that the information basis for my design effort widened at the same time.
7. In parallel with point 6 I made my first practical attempt to use Deft with the Gane-Sarson design and description technique to transform the explicit wall graphs into data flow diagrams, but I was not ready for it on either the methodological or the practical level. I recognized very clearly that I should understand the situation to be designed thoroughly before I could go into the descriptions any further.
8. In parallel with point 6, I recognized that the ER-methodology should be used for data design and conceptual schema descriptions when analyzing the descriptions of chapter 3 of the IFIP case. This implied the same kind of self-teaching and learning process as with the Gane-Sarson methodology.
9. I found during our team meeting that there was a large gap between my and that of the others understanding of the concepts of thrownness, breakdown, readiness-to-hand and creativity. This led to a new team learning process in order to achieve a more homogeneous conceptual understanding. I felt that this was very important.
10. I returned to using the use wall graph technique, and made a new wall graph based on the previous two, the text of chapter 3 of the IFIP Case and the literature concerning the general principles of materials handling in a typical organization. After finishing this wall graph I felt ready to make my first data flow diagram using Deft.
11. I managed to make the first version of a Gane-Sarson data flow diagram with Deft, a version that reflected my impression of the materials handling process of the company where the application under design is imagined as being used. I was not satisfied with the contents of the diagram and I was ready to make new versions by utilizing the content of chapter 3 of the IFIP Case to a greater extent. I had problems in following the Gane-Sarson methodology during this design and description process e.g. when sketching the diagram before actually drawing it using Deft. I was thus left with an impression that I had to return to making manual sketches before I could continue with Deft descriptions. I also had problems with the actual drawing dialogue with Deft, because I could not see the whole diagram at the same time as I was drawing it. On the other hand, Deft followed general guide-lines typical to all Mac software, and was in this sense easy to use and user-

supportive.

12. Excellent ! I succeeded in designing and describing six versions of the Inventory Control and Purchase system using the Gane-Sarson methodology and Deft, and also employed external material supporting the IFIP Case description. I followed the Gane-Sarson methodology and identified all the basic external entities and basic material flows at first. I then drew them using Deft, printed several copies of the description and sketched initial versions of the diagrams on them manually before drawing these with Deft. I iterated all the versions and compared them with the early wall graphs. All the new versions included more precise designs and descriptions than the early versions. Finally I combined all the exploded processes and designed and described my final Gane-Sarson data flow diagram for the Inventory Control and Purchase system. The next step was to continue into data design using the ER methodology and Deft.
13. The information content of chapter 3 of the IFIP Case forms a quite complete basis for the data dictionary definitions. At the same time Deft offers marvellous support for transformation into the ER methodology and descriptions.
14. No self-reflections during the experiment process.

Subjective conclusions

1. Effects of the Case tool on the creativity of the Information System Producer (designer).Two levels of creativity can be recognized in the producer role, creativity with the use of the methodologies included in the CASE tool, and creativity with the use of the CASE tool itself. In order to be creative in the design process, the first treshold is to master both the methodologies and the behaviour of the CASE tool on a level that means that you are able to do design work with them. This depends more on the ability of the designer to be creative, although it also depends on the features of the CASE tool. The second level of creativity consists of the ability of the designer to avoid all the features that prevent creativity or deficiencies of the tool or the methodologies included in it. Purely tool-based creativity support does not exist, because creativity is always connected with the subject designing the system. Thus the creativity support of a CASE tool in general is dependent on the relationship between the producer and the tool.

2. Effects of the Case tool on the creativity of its consumer. I see that in my experiment the case tool both increased and detracted from my creativity during the systems design process. The increase was based on the possibilities to generate, investigate and document new versions of the specifications easily, while the CASE tool was also preventing me from being creative, because it could not support the methodologies with which I was already acquainted. Because this experiment was also my first contract with the use of the CASE tools, it was a learning process rather than a real designing or specification process, and consequent the case tool had more of a

restraining than a promoting effect on my personal creativity.

3. General remarks. I felt the use of the creativity and kernel concepts to be quite problematical during the whole experiment. This may reflect my own personality, in that I could not recognize my internal states very easily through the kernel concepts. Instead I saw myself alternating dynamically between these states when performing a certain subactivity, e.g. the collection of verbs describing acts of creativity as defined by Shore (Shore 1977). Thus, all of the creativity findings during my experiment should be understood as recognition of the dominant state during a creative activity connected with an identified object of interest.

4.3 Experiment C

Designer in this experiment described himself by the MBTI self-assesment as a combination of assimilator and converger.

<u>Experimental data</u>

Steps -Objects of Interest	Kernel concepts and Creativity	Thrownness	Ready-to-hand	Breakdown	Blindness	Language	Recurrent patterns	Creativity
Step A	Whole experimentation	1		1				
Step B	IFIP-case description and first design effort		2	3		2		4
Step C	Intermediate reporting			5				5
Step D	Individual reflections							
	Thrownness	6		6				
	Breakdown			7				
	Blindness			8	8			
Step E	Detailed planning		9		9		9	9
Step F	DEFT grasping			10	10			
Step G	Team meeting			11				11

Table 3

1. Our process of experimentation includes purposive breakdowns of practical specification work – prohibiting the natural thrownness! The purposiveness lies in the shift of our

domain of concern from practice to creativity components as research objects and vice versa. This is an artificial situation in the sense of its environmental context, but individually I feel committed and deeply interested in the experiment.

2. The following subactivities should be implemented in the specification process:
   - identification of all the interest groups (IG)
   - state, future and problem analysis of each IG
   - goal, constraint and effect analysis of the change for each IG
   - goal, constraint and effect synthesis
   - design of implementation and test

   The 'rich picture' has necessarily to be developed using manual sketching and/or wall graph techniques — or Deft, following the problem set for this experiment!
3. Since I had no earlier experience with Deft, the main problem was how to implement the descriptions needed in subactivities with Deft, and before that, how to learn feasible and suitable features of Deft for this situation.
4. I used ERD for IG activity analysis, and especially problem (recurrent breakdowns), which was feasible even without the guidance of Deft ERD documentation. Observe that 'creativity' in this step concerns the selection of the description language, not the creative design (specification) of domain knowledge ideas!
5. The preunderstandings and interpretations of thrownness and breakdown achieved by subjects B and C differ too much. How can it be that we are approaching these design problems so differently? Team creativity is challenging, but are our team meetings preventing or supporting it?
6. A better understanding of this concept:

   Level of commitment to the situation on the basis of environmental and individual emphasis (being 'thrown') – I have earlier emphasized individuality too much!
7. Breakdown = unreadiness-to-hand, surely! It creates reflective observations.
8. Reflecting creates blindness, naturally and necessarily, based on our language skill — and experiential style, too! Our team is rich in its variety of styles.
9. A detailed plan to implement my specification process was the result. I have to be sure, before making any concrete design effort, how my effort will emphasize new creative features of the existing IFIP case, especially on the IS end user level. Personally, it is natural to me that 'deep planning always comes before implementation'. This repeats my existing capability, however, and shows 'personal uncreativity' in this practical effort.
10. I am apt to be creative in relation to the domain knowledge of this specification problem, but uncreative with regard to our description tool and language. Why? My progress is slow in any case!
11. I'm highly interested in the style information of the other team members! A hypothesis: "Team creativity can be implemented by use of a 'style-variety' team in which individuals

act 'uncreatively' – i.e. repeating one's style capabilities! But then the question arises: "What kind is team creativity if each member really behaves creatively?

Subjective conclusions

Creativity lies in the rapports between an individual and his/her activity environment: design tasks, CASE tools and other human participants. These rapports vary with individual experiential styles and manifest themselves in the form of sensitivity to breakdowns (and blindness because of that) and in ability to develop recurrent patterns in action.

We can characterize three subconcepts – different aspects of creativity:

- design task (and CASE tool) creativity:
  a process in which participating individuals produce new, original ideas for the purposes of a given task with or without the tools – a CASE tool can support or prevent this depending on its user; the individuals can act uncreatively or creatively.The main problem, however, is what are the rapports between the task demand, CASE tool characteristics and end user ability (cf. comments 3, 4 and 9),
- individual creativity:
  in the context of her/his dominant style, an individual is aware of and able to reduce his blindness and to add skill with language; individuals can occupy different roles in the specification process (cf. comments 5, 6, 8 and 10),
- team creativity:
  the collaborative ability of the committed group of individuals (design team) to develop original ideas more effectively than the sum of individual abilities. Because of the polylectic features of different basic styles, the existence of team creativity is highly variable and is in a continuous state of change (cf. comments 5 and 11 and the next section below).

(In the sense of these evaluations the dear reader is encouraged read the style characterizations and the experiments once again!)

5. CONCLUSIONS

We tentatively deduced that creativity is the special subjective ability which is achieved within thrownness, is based on readiness to hand, happens during breakdowns concerning that particular breakdown and other future ones, is implemented by language and prevented by blindness. To increase creativity means to motivate ourselves to increase our language skills and reduce our blindness and consciously directed evolution of recurrent patterns in design in order to release human resources for idea generation. The human experiential styles are context factors of creativity, influencing the selection of motives, objects of interest and operations. Different kinds of creativity probably appears in the context of different styles. The features of non-dominant basic styles can be the essential sources of personal creative development for the individual.

Creativity lies in rapports between an individual and his/her activity environment: design tasks, CASE tools and other human participants. These rapports vary in individual experiential style and manifest themselves as sensitivity to breakdowns (and blindness because of those) and to the ability to develop recurrent patterns in action.In this study the research team showed a remarkably wide variety within the polylectical Kolbian field. This apparently gives one explanation for the different, individually characteristic, approaches they used in the experiments.

Based on the experiments we can characterize three aspects of creativity:

> Design **task** (and CASE tool) **creativity**; a process in which participating individuals produce new, original ideas for the purpose of a given task with or without CASE tools – the main problem is what are the rapports between the task demand, CASE tool characteristics and end user ability
>
> **Individual creativity**; in the context of her/his dominant style, an individual is aware of and able to reduce his blindness and to increase his skill with language
>
> **Team creativity**; collaborative ability of a committed group of individuals (design team) to develop original ideas more effectively than the sum of individual abilities.

The following, partially paradoxical process exists in the individual evolutionary process for using the CASE tool :

- the CASE tool prevents creativity during the **learning** phase, when a designer get acquainted with the CASE tool, by binding designer's human resources,
- the CASE tool supports creativity during the **inspiration** and **maturation** phases, when a designer has sufficient ability and expertise for using the CASE tool and avoiding it's restrictions in a creative way,
- the CASE tool either supports or prevents creativity during the phase of **routine** (and boredom), depending on the awareness of the designer in understanding these routines as a way of releasing resources for creative effort somewhere else.

The research process implemented here was felt by all the participants to have been creative, collaborative and interesting, but at the same time fuzzy, divergent and highly demanding. A little more structure and more homogeneous working habits would be preferable in future efforts. The most appropriate collaborative team effort for CASE tool-supported systems design was lacking here, but would be the natural next object of research.

REFERENCES

Bubenko 1988	Bubenko Janis A. jr., Selecting a strategy for computer-aided software engineering (CASE), p. 23, SYSLAB University of Stockholm, Stockholm, June 1988.
Case 1986	Case Albert F., Information systems development: principles of computer-aided software engineering, Prentice-Hall, Englewood Cliffs, New Jersey 1986.
Chikofsky et al.1988	Chikofsky Elliot J. and Rubenstein Burt L., CASE: Reliability Engineering for Information Systems, IEEE Software, pp. 11 - 16, March 1988.
Deft 1988	Deft Infotool Manual Version 3.1, Deft Inc. 1988.
Fairley 1985	Fairley Richard E., Software Engineering Concepts, McGraw-Hill Book Company, USA 1985.
Fisher 1988	Fisher Alan S., CASE - Using Software Development Tools, John Wiley & Sons, USA 1988.
Gane et al. 1979	Gane C. and Sarson T., Structured Systems Analysis: Tools and Techniques, Prentice-Hall Inc., New Jersey 1979.
Huling 1987	Huling Jim, Tools of the trade: Is CASE really a cure-all?, Computerworld, April 1987, pp. 73 - 86.
Kolb 1984	Kolb D., Experiential Learning - Experiences as the source of learning and development, Prentice-Hall, 1984.
Kumar et al. 1988	Kumar Kuldeep and Welke Richard J., Methodology Engineering: A Proposal for Situation Specific Methodology Construction, in Proceedings of CASE Studies 1988, Meta Systems, Ann Arbor, 1988, Meta Ref. #C8811.
Mathiassén 1988	Mathiassén Lars, Creativity and Discipline in Systems Design, Institute of Electronic Systems, University of Aalborg, Denmark 1988.
McClure 1988	McClure Carma, The CASE Technical Report, Extended Intelligence Inc., Chicago 1988.
Myers et al. 1962	Myers and Briggs, Type Indicator Manual, Princeton, New Jersey, Educ. Test. Serv., 1962.
Olle 1988	Olle T.W., Business Analysis and System Design Specifications for an Inventory Control and Purchasing System in Computerized Assistance During the Information Systems Life Cycle: Proceedings of the IFIP WG

	8.1 Working Conference on Computerized Assistance During the Information Systems Life Cycle, CRIS 88, Egham, England, 19 - 22 September, 1988, ed. T.W. Olle, A.A. Verrijn-Stuart and L. Bhabuta, pp. 463 - 496, North-Holland, Amsterdam 1988.
Shore 1977	Shore S., Creativity in Action, Sharon Corp. 1977.
Stefik et al. 1987	Stefik et al., Reviews on the book by Winograd and Flores, in Artificial Intelligence No 31, 1988, pp. 213 261.
Wallas 1926	Wallas G., The Art of Thought, Harcourt Brace, New York 1926.
Windsor 1986	Windsor John C., Are Automated Tools Changing Systems Analysis and Design, Journal of Systems Management, November 1986, pp. 28 - 32.
Winograd et al.1986	Winograd T. and Flores F., Understanding Computers and Cognition - a new foundation for design, Ablex Publ. Corp., New Jersey 1986.
Yourdon et al. 1979	Yourdon E. and Constantine L.L, Structured Design: Fundamentals of a Discipline of Computer Program and Systems Design, Prentice-Hall Inc., New Jersey 1979.

FLAWS IN RATIONALISTIC THINKING, EXACT CREATIVITY AND INFORMATION SYSTEMS

Kari Kuutti
Dept. of Information Processing Science, Univ. of Oulu
Linnanmaa, SF-90570 OULU, FINLAND

Abstract: The creativity phenomenon is studied against the background of the ongoing epistemological debate in the area of Information Systems research.

A hypothesis is put forward, that the utilizing of creativity - as it is now generally understood - is an unintentional and unsystematic attempt to overcome the epistemological problems, which have been criticized in the dominant research paradigm.

An alternative view on creativity is presented, based on one of the challenging epistemologies, namely dialectical materialism. The view is concretized by taking the book *Creativity as an Exact Science* (Altshuller 1988) as an example. The main conclusion is that the key to the success lies *in the understanding of the nature and dynamics of the system to be changed.* Creativity techniques are unsystematic, blind attempts to overcome the lack of such an understanding.

Finally some additional discussion is entered into on whether we really need the concept of creativity. It is suggested that it should be replaced by "expansive learning".

1. Introduction

When the terms "creativity" and "information systems" (IS) are brought together, this sounds very natural at first. Why not? Hardly any phenomenon has gained such overwhelming popularity recently as "creativity", which has been proposed as a patent cure for almost everything problematic between heaven and earth. There are a growing number of popular books which address the creativity problem, creativity training has gained a steady position among the variety of different consultancies, and there is a continuing discussion on how creativity should be taken into account even in schools and universities. Naturally it has also penetrated into the area of information technology, as can be clearly seen in recent advertising, where every computer and program - albeit the dullest PC clones or most boring spreadsheets - claims to support creativity. So, why should there not be some connection also with information systems?

One begins to wonder however, why this question should have arisen just now? What is the reason behind the eagerness to search for connections between IS and creativity, as in this IRIS conference? Is it caused by some recent advancement or breakthrough in our understanding about creativity?

No, definitely not. Although research into creativity is continuing, there is no remarkable progress in sight. One can't help the feeling that the most influental work was done 20, 30, or even 40 years ago by such people as Guilford, Wertheimer, Bartlett, Torrance etc., and since then the same ideas have just been warmed up and polished under different headings. In fact, the progress may be going in reverse for it seems impossible to find any recent attempt to match the classics such as Wertheimer (1945) in richness of thought.

If there is nothing new on the creativity side, the reason for its actuality has to be found on IS side, where there really is an epistemological debate going on between the dominant research paradigm and its challengers - a debate to which creativity research could perhaps contribute. This seems to be the most understandable reason for the growing interest in this concept among IS researchers. Thus the creativity phenomenon has to be studied against the background of that epistemological debate.

A hypothesis is put forward here that the utilization of creativity - as it is now generally understood - is an unintentional and unsystematic attempt to overcome the epistemological problems which have been criticized in the dominant research paradigm. This hypothesis is studied in section 2.

An alternative view on creativity is then presented in section 3 which is based on one of the challenging epistemologies, namely dialectical materialism. This view is concretized by taking the book *Creativity as an Exact Science* (Altshuller 1988) as an example.

Section 4 then contains some additional discussion on whether we really need the concept of creativity, and it is suggested that it should be replaced by "expansive learning".

2. Creativity and the paradigmatic debate within the IS research community.

2.1 Paradigmatic discussion

One of the main features of recent research in the area of IS is the conflict between the established dominant paradigm and several emerging new challengers. This conflict is thought to be a result of the ongoing change in society at large (see Klein & Hirschheim 1987, Kuutti 1989), and can also be seen in the framework of the broader debate about the paradigms of the social sciences in general (a good reference to that discussion can be found in Reason & Rowan 1981). The debate within the social sciences first penetrated into the organizational sciences and from there into IS research, influenced to a major degree by the seminal book of Burrell & Morgan (1979). The conflict and the existence of several competing research paradigms was clearly manifested in the "Manchester colloquium" of 1984, the results of which are published in Mumford et al. (1985).

What is the dominant paradigm, which is known variously as the "rationalistic tradition" (Winograd & Flores 1987), "positivist science" (Hirschheim 1985), "scientism" (Klein & Lyytinen 1985) or the "positivistic-empiristic approach" (Harré 1981)? It can be determined as follows:

"For the purpose of my discussion, positivism will be summarized as being based on five pillars:

1) Unity of the scientific method.

2) Search for Humean causal relationships.

3) Belief in empiricism.

4) Science (and its process) is value-free.

5) The foundation of science is based on logic and mathematics." (Hirschheim 1985, p. 16).

There are numerous papers which cast doubts on these principles as guides for IS research, the main direction in the criticism having been that information systems are social systems and that social phenomena cannot be adequately studied using a positivistic approach.

Although it has not been the main stance, there are also opinions which blame the positivistic approach for being too rigorous and mechanistic:

"But it [naive inquiry, KK] also has a lot of very good qualities, because it is involved, committed, relevant, intuitive; above all it is *alive*. So this kind of naive inquiry is a very important part of our humanity, it is what we all start with, and we lose a lot if we try to throw it out altogether.

But this is what so-called objective research does: in order to get away from the subjectivity and error of naive inquiry, the whole apparatus of experimental method, quasi-experimental method, statistical significance, dependent and independent variables, and so on is set up. While this does counter some problems of naive inquiry, it also kills off everything it comes into contact with, so what we are left with is dead knowledge." (Reason & Rowan 1981, p. xiii).

"Information systems development and use involves people in action. Still information systems research is dominated by ideals largely fetched from Newtonian physics. This leaves no room for human innovative action." (Nissen 1985, p. 39).

2.2 Creativity research

Creativity has been found to be a multifaceted phenomenon. It is widely accepted that four major aspects can be distinguished: 1) the creative process, 2) the creative product, 3) the creative person and 4) the creative situation/environment. Correspondingly, there are different research approaches, anwering different questions:

> "(1) What is the nature of the creative process? What are the qualities and kinds of psychological processes by which creative solutions to problems are achieved? (2) What are creative products? By what qualities they can be identified? (3) What are the distinguishing traits and characteristics of creative persons? (4) What are the specifications of the creative situation, the life circumstance, or the social, cultural, and work milieu which facilitate and encourage the appearance of creative thought and action?" (MacKinnon 1978, p. 46).

Although there had been a lot of personality studies earlier, a considerable increase in interest in other facets arose in 1950s. Creativity research has been most popular in the U.S.A., and a number of research bodies such as the*Institute for Personality Research and Assessment* and the *Creative Problem-Solving Institute* have been founded. There is also a scientific journal, the *Journal of Creative Behaviour*. Creativity training in different forms has gained wide acceptance in industry.

Perhaps the most popular - and constantly relevant - research topic has been the nature of the creative process and how to support it. This is understandable, as any advances in understanding could be of immediate benefit and economic advantage. There exists at least a vague consensus over the main features of the creative process.

Firstly, it is fundamentally a problem-solving process, or can be described as such. "The creative process starts always with the seeing or sensing of a problem." (MacKinnon 1978, p. 47). "No creative thinking or 'satori' is likely to occur until there is a recognition or awareness of the problem and there is some definition of the problem and commitment to deal with it." (Torrance 1979, p. 13).

Secondly, the process can be separated into certain steps, e.g. 1) period of preparation, 2) period of concentrated effort, 3) period of withdrawal, 4) moment of insight and 5) period of verification (MacKinnon 1978, p. 47).

Thirdly, there are different forms of supporting technique for each of the steps. The most popular form of support is to help the generating of more ideas: "There has been considerable demonstration that the more alternatives a person or group produces and considers, not only the more viable those solutions are likely to be, but also there is a greater likelihood of success in solving problems." (Torrance 1979, p. 24).

Fourthly, creative thinking is spontaneous and child-like, free from the restrictions of "normal, critical, analytical thinking".

> " ..., yet paradoxically, efficient, economical, and analytical perception are sometimes the enemy of creative insight. Analysis disassembles a whole into its parts, separating out from one another the elements of a problem. At a certain stage this is necessary, if progress is to be made; but in the course of analyzing a problem, certain attributes which pertain to the phenomenon as a whole may be destroyed with the danger that eventually one 'cannot see the woods for the trees'. What is then needed, if there is to be a creative reorganization, is a compensating, free, spontaneous look at the whole situation, a naive and childlike apprehension of what is there." (MacKinnon 1978, p. 49).

Fifthly, there is no dominant theory concerning the psychological processes which underlie creativity. One of the most popular ones uses the "conscious -subconscious" concepts by Freud for explanatory purposes.

2.3 Discussion

Although the creativity approach is practically oriented and does not pose epistemological questions, it is clearly in intellectual opposition to rationalistic thinking. Of the "pillars" of Hirscheim, it challenges at least the scientific method, empiricism and mathematical-logical foundation, and in doing this it comes very close to the criticism presented by the "anti-positivists". On the other hand, it sees its position more complementary rather than a contradictory one.

I believe, that it is not too bold an interpretation to say that the recent interest in "creativity" is in fact *an unconscious and unsystematic approach intended to "patch" flaws in rationalistic thinking.* It has had some success in industrial practice, because in that field it is not possible to overcome problems just by defining them "unscientific".

One can perceive a peculiarity in the situation. There is a internal contradiction in positivistic research, in that it is totally impossible to do any research at all using only the rigorous principles of "scientific method", because both the stating of problems and the interpretation of results lie beyond its reach. This is also recognized by the critics:

> "From this one must conclude that by relying on human consensus for the interpretation of data, scientistic research violates its own standards of the objectivity and rigor. All of a sudden reflection and speculation are allowed to enter." (Klein & Lyytinen 1985, p. 139).

Thus there is plenty room for creativity in the problem-setting and result interpretation aspects of rigorous, "scientific" research, even though these areas have not been specifically explored by the creativity research, which has tended to challenge the "scientific" thinking in its core area, namely problem-solving.

3. Dialectical thought and exact creativity.

There is one approach which has not yet contributed to the "positivistic - antipositivistic - creativity" debate, namely that of "traditional" marxism. The following is an attempt to initiate the rectifying of this shortcoming.

3.1 Engels on creativity

One of the most important problems in creativity is - in the terms of the previous chapter - the lack of spontaneity in thinking. The most important and most numerous tools and techniques are aimed at greater freedom of thinking. Let us now take a different viewpoint in the form of the well-known passage from "Anti-Dühring" (Engels 1962).

> "Hegel was the first one to state correctly the relation between freedom and necessity. "Necessity is *blind* only *in so far as it is not understood.*" Freedom does not consist in the dream of independence from natural laws, but in the knowledge of these laws, and in the possibility this gives of systematically making them work towards definite ends. This holds good in relation both to the laws of external nature and to those which govern the bodily and mental existence of men themselves - two classes of laws which we separate from each other at most only in thought but not in reality. Freedom of the will therefore means nothing but the capacity to make decisions with knowledge of the subject. Therefore the *freer* a man's judgment is in relation to a definite question, the greater is the *necessity* with which the content of this judgment will be determined; while the uncertainty, founded on ignorance, which seems to make an arbitrary choice among many different and conflicting possible decisions, shows precisely by this that it is not free, that it is controlled by the very object it should itself control. Freedom therefore consist in the control over ourselves and over external nature, a control founded on knowledge on natural necessity; it is therefore necessarily a product of historical development." (p. 157).

How can one then achieve "freedom of thinking" - deep understanding? According to marxism, one of the strongest means for doing this is the use of dialectical thinking. The term dialectics in the marxist context should neither be confused with formal dialectics in logic or with the using of dichotomies. It is better characterized as an attempt to develop a general theory of development.[1] "Dialectics, on the other hand, comprehends things and their representations in their essential connection, concatenation, motion, origin and ending." (Engels 1962, p 36). One of the key elements of this theory is the perception of contradictions as one dominant force behind development:

[1] Readers with a deeper interest in marxist dialectics should consult a general review such as that of Konstantinov et al. (eds.) : *The fundamentals of the marxist-leninist philosophy*, Progress Publ. Moscow 1982, a specialized work on the topic, like Ilyenkov: *Dialectical logic: Essays on its history and theory*, Progress Publ., Moscow 1977, or a classic, such as *Anti-Dühring* by Engels cited in this paper.

"True, so long as we consider things as at rest and lifeless, each one by itself, alongside and after each other, we do not run up against any contradictions in them. We find certain qualities which are partly common to, partly different from, and even contradictory to each other, but which in the last-mentioned case are distributed within different objects and therefore contain no contradiction within. Inside the limits of this sphere of observation we can get along on the basis of the usual, metaphysical mode of thought. But the position is quite different as soon as we consider things in their motion, their change, their life, their reciprocal influence to one another. Then we immediately become involved in contradictions." (Engels 1962, p. 166).

3.2 The exact theory of creativity of Altshuller

As an example, we can study a different approach, in which the principles presented at the general level in the preceding section are put to work. This approach was developed by a Russian engineer G.S. Altshuller, and his collagues, the main goal having been to develop support for technical inventions.

Altshuller's approach differs radically from popular creativity support in many ways:

- Where popular creativity support is "against method" and makes claims like "rational thinking kills creativity", Altshuller's approach is quite systematic and aims at rationality. His collection of support tools is even called "The algorithm to make technical inventions", which may irritate anyone with a computer science background. He means "heuristics" in our terminology, of course.

- Where the usual claims are that "knowledge kills creativity" and "the less you know about the subject matter, the more creative you can be", Altshuller takes exactly the opposite point of view. "Freedom lies in the understanding of the true necessities"; you are free to select only if you know all the true alternatives. Altshuller's approach is aimed at professionals: the more knowledgeable a professional user is, the better are his chances of producing something new.

- Where the popular trend is towards "Quick-Aid" for creativity, Altshuller's approach does not attempt to be easy to follow or master. In fact, several weeks or even months of active training are needed before one can expect to gain any benefit (Rantanen 1989).

- Where the popular creativity support approaches claim to be more or less universal and suitable for every area, Altshuller's approach is strictly bound to one subject so far - the mechanical engineering.

The main idea of the approach is quite old and simple: model a problem, resolve the model and apply the solution back to the reality - traditional rational thinking with models. The nature of the phenomenon to be modelled is quite extraordinary, of course, and will need its own conceptual tools. Altshuller has the view that technological systems evolve through "stages", each more advanced than the preceding one. Development occurs when a contradiction in the system is resolved. The stages, contradictions and ways of resolving them can be found by analyzing the existing patent information in which successful results of problem-solving events are documented.

Roughly, the "algorithm" proceeds in the following way (Fig. 1). Firstly, a highly abstract model is formed for the technical system and its fundamental contradiction . The model has no external resemblance to the original system and very different systems can be described with the same model. Secondly, the contradiction is removed by following rules derived from the patent information. Thirdly, the model solution is realized. One can get some help from examples, which describe similar model solutions realized (and patented) earlier.

As we can see, the "creative act" has not faded away. It is still there, inside the "realization" phase. There the inventor has to use all his/her professional expertise and creativity in order to imagine what the highly abstract model solution to the model could mean in reality. The model gives only "a high-level hint" of where a good solution could be found. Whether it can be found, and what it will be like will depend of the abilities of the inventor.

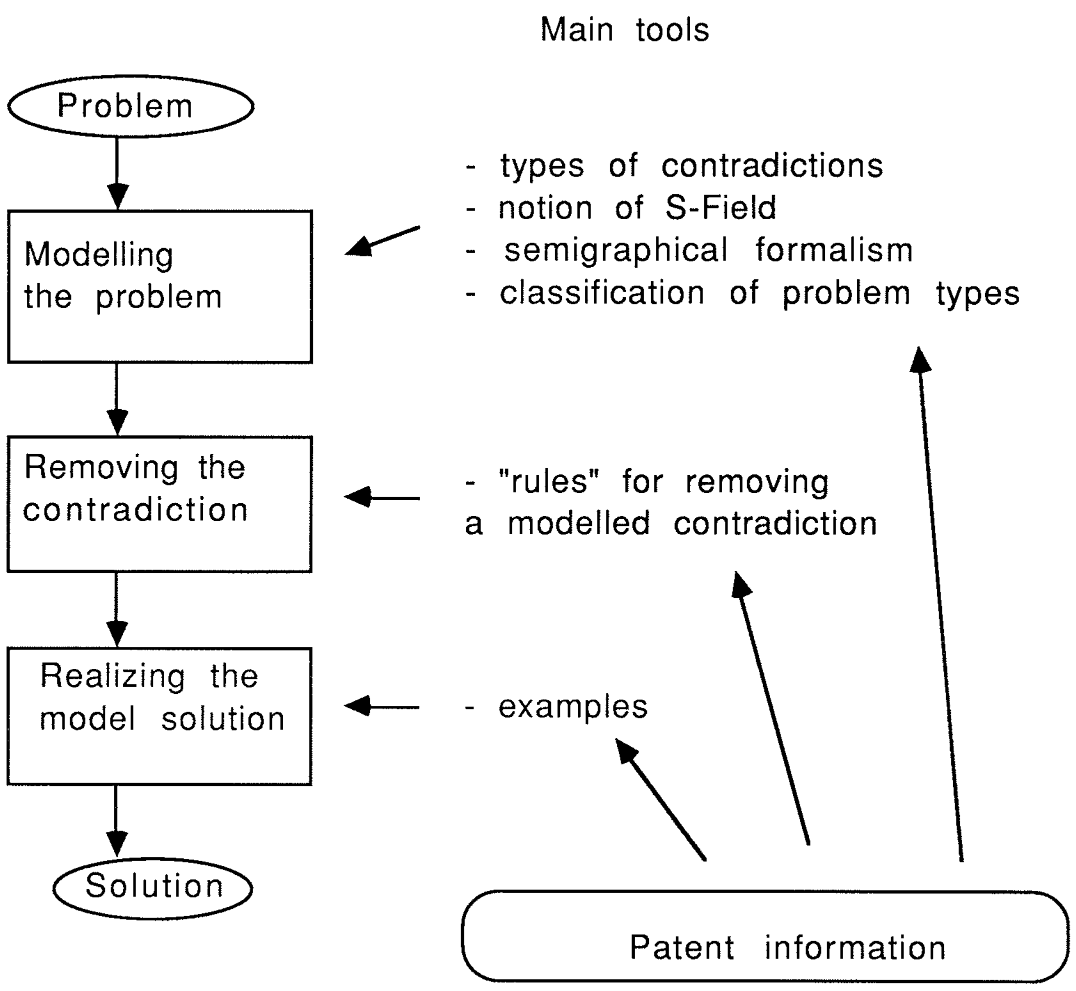

Main tools
Problem
Modelling the problem
Removing the contradiction
Realizing the model solution
Solution
- types of contradictions
- notion of S-Field
- semigraphical formalism
- classification of problem types
- "rules" for removing a modelled contradiction
- examples
Patent information

Figure 1.

The main idea of the approach can be condensed as being to collect and use a new type of knowledge - knowledge about the "genetic" development of technical systems - in order to give not solutions, but good advice on where solutions could be found.

3.3 A more detailed look at the Altshuller approach

The following section puts forward a slightly more detailed outline of the theory.

Altshuller criticizes research on creativity:

> " Psychologists started to experiment with simple tasks. (...) It emerged that the experimenters solve tasks by compiling a list of variants, that in this process much depends on their previous experience and that each variant considered transforms their view on task, etc. However, this this did not shed any light on the main problem of why do certain inventors manage to

solve a problem with a small number of trials when it is generally thought that they will need a larger number of trials?

The psychology of creativity cannot answer this question to this day. In essence since the 1930's and 1940's no new results in principle have been obtained." (Altshuller 1988, p.6).

The cause of problems in studying creativity is that there is no such thing as 'creativity in general', but instead creative problem-solving is always connected with some particular subject matter, and that is the primary factor in the process. Thus however efficiently we support the psychological aspects of the creative process, our success will be very restricted unless we understand the application area and its laws of development. "Freedom of the will means nothing but capacity to make decisions based on the knowledge about the subject".

" However, the psychological factors are secondary and conditional. The main thing in invention is that the technological system is transferred from one state to another, and according to definite laws and not 'just at random'. But precisely this is the primary and objective side of the creativity which remains beyond the horizon of psychologists.

Imagine that we are studying the actions of a helmsman aboard ship on a meandering river. We want to know nothing about the river itself but only try to explain the actions of the helmsman in purely psychological terms. We see the helmsman beginning to spin the wheel rapidly to the right. Why? Maybe the sun is in his eyes and he is turning away from it - that is the reason... But now he is slowly turning the wheel to the left. Why? Perhaps he has decided to turn his face to the sun after all and get a tan? But now the helmsmen have changed over and the new man at once begins to turn the wheel and - Great God - he has turned his back to the sun. All right, this means that the behaviour of helmsmen is determined by whether or not they like sunbathing, so let's note it down...

Unfortunately this is no exaggeration: this is precisely how the purely psychlogical approach, ignoring the existence of objective laws of development of technical system, looks." (Altshuller 1988, p. 8-9).

The key element of the theory is - in total harmony with dialectics - the hypothesis that technological systems will evolve according to some quite stable and general laws and that it is also possible to study those laws. Compared with the common view, this is a revolutionary statement, but it does not as such give much help in practice. If practical advice is wanted, one has first to determine these laws.

Naturally it is impossible to just 'invent' the laws. They can be found only as a result of long and laborious research, and that is just what Altshuller and his colleagues have been doing for the last 30 years. What makes the effort meaningful is the abundance of existing information about innovations, namely that collected in the form of patents. The first thing was to form a classification for innovations:

> "If you were to ask someone how to hunt, you would immediately be asked in return what you wanted to hunt. Microbes, mosquitos, whales are all living creatures which can be hunted. But hunting for microbes, mosquitoes and whales takes on three qualitatively different forms. No one would study these three forms 'in general'. In invention, however, for a long time creativity was studied 'in general' and conclusions drawn from 'microbe' inventions were widely applied to 'whale' inventions, and vice versa." (Altshuller 1988, p.16).

Altshuller classifies innovations into five categories according to the change involved:

1) The object (device or method) does not change.

2) The object is changed, but not substantially.

3) The object is changed substantially.

4) The object is changed totally.

5) The entire technological system into which the object fits is changed.

> "Solution of a problem of the first sort requires choosing between a few obvious variants. (...) On the second level the number of variants is already measured in dozens. (...) A correct solution of problems of the third level may be buried

among hundreds of incorrect ones. Thousands and tens of thousands of trials and errors can be made on the fourth level in order to find a solution. Finally, on the fifth level, the number of trials and errors grows to hundreds of thousands and millions." (Altshuller 1988, p.21).

> "The higher level problems differ from the lower not only in the number of trials it takes to find a solution. There is also a qualitative difference. The problems of the first level and the means of solving them are to be found within the confines of one narrow speciality. Problems of the second kind and the means of solving them relate to a single field of technology. For solving problems of the third level, one has to turn to other fields. The solution of the problems of the fourth level must be sought not in technology but in science, usually among little utilized physical and chemical effects and phenomena. At a higher sub-level, means of solving problems of the fifth level may in general turn out to be beyond the limits of contemporary science. Therefore first one must make the discovery and then, relying on the new scientific data, solve the invention problem. (...) The higher level the broader the knowledge called for." (Altshuller 1988, p 23-24).

The most interesting cases are those on the highest level, the study of which can reveal some of the secrets of technical innovations.

Contradictions: the core of problems

Problems as such are not yet suitable for analysis, because they are too specific and particular. One has to have better conceptual tools for generalizing the situation. Altshuller uses the concept of contradiction, which he determines at three levels of depth.

> "In point of fact a contradiction is already present in the origin of inventive problems. Something has to be done, but how to do it is unknown. Such contradictions are customarily called 'administrative' (AC). There is no need to discover administrative contradictions since they lie at the surface of the problem. But the heuristic prompting force of such contradictions amounts to nil. They do not say in what direction the solution should be sought.

Below the administrative level lie the technical contradictions (TC); if by certain methods one improves one part (or one parameter) of a technical system, it is inadmissible for another part (or another parameter) to deteroriate in the process. (...) ... correctly formulated TC possesses a definite heuristic value. (...)

Each TC has specific physical causes. (...) This is a physical contradiction (PC): mutually opposing demands are placed upon one and the same system.

In physical contradictions the conflict of demands is intensified to the maximum. Therefore at first glance the PC seems absurd, inadmissible by definition. (...) But it is in precisely this, carrying the contradictions to the extreme, that the heuristic strength of the PC shows itself." (Altshuller 1988, pp 28-30).

S-Field: a tool for modelling problems

There also exists a need to model and handle problems and contradictions in a convenient and compact way. Altshuller has developed a theoretical construct for that purpose.

"The two substances and a field can be completely dissimilar, but they are necessary and sufficient for the formation of a minimal technical system which has been given the name S-Field (from Substance and Field).

In introducing the S-Field concept we utilize three terms: substance, field, mutual interaction (effect action, connection). By the term 'substance' we understand any objects no matter what their degree of complexity. Ice and ice-breaker, a screw and a nut, a cable and a load - all of these are substances. Mutual interaction is the universal form of bodies and phenomena, resulting in their mutual change. (...) We shall use the term 'field' in a very broad sense, and together with the "legitimate" physical fields regard all possible kinds of "technical" fields - heat, mechanical, acoustic, etc. as such." (Altshuller 1988, pp. 52-53).

Methods and laws: tools for handling dynamics

It is not enough to recognize and model the problem in the terms of contradictions, but it should also be solved. For that purpose, it is necessary to know the possible ways, in which a technical system can move from one state to another:

> "Where can one take the assemblage of methods sufficiently rich in order to solve the most varied inventive problems? The answer is obvious: physical contradictions are inherent only to inventive problems of the highest levels, therefore methods for eliminating them should be sought in the solutions of these problems."

By analyzing the most inventive solutions among patent information (about 40 000 patents have been studied thoroughly), Altshuller and his colleagues have collected a selection of "strong" methods. As such they are already helpful: one can universalize them by the means of a S-Field analysis and then transfer the essentials of a solution to another, perhaps totally different area. More can still be done, however, for by studying the history of different technical systems one can determine in which order the different types of solutions are most likely to emerge, and thus find "laws" of development.

> "A good list of methods of removing the PC alone would be a lot in itself. But one has to have correctly formulated contradictions and also know when and what method to use; one must be in posession of criteria for evaluating the results one has obtained. For this it is necessary to know the rules of development of technical systems.
>
> The development of technical systems, like all other systems is subject to the general laws of dialectics. In order to concretize these laws applicable precisely to technical systems one has to turn once again to the patent storehouse but going now to a considerably greater depth." (Altshuller 1988, pp 31-32).

ASIP: an "algorithm for inventions"

Recent knowledge on the achievement of technical innovations may be condesed into an "algorithm" (named ASIP-77), a description of the steps and tools needed in order to approach a technical problem systematically. ASIP may be outlined as follows:

1) Selection of the problem.

2) Construction of a model for the problem. (The S-Field construct is used to find the physical contradiction. There is also a semi-graphical formalism to help in modelling.)

3) Analysis of the model. (The contradiction is recognized, classified and its place in the development of that particular technical system located.)

4) Removal of the physical contradiction. ("You have this and that kind of contradiction. In this and that kind of situation the technical system generally develops so, that this form of relation is replaced with a more advanced one, usually with one of this type." - The "invention" still has to be made as to how the general rule is realizable in this situation. Thus ASIP is not a real algorithm in the strict sense.)

5) Preliminary assessment of the solution obtained.

6) Development of the answer.

7) Analysis of the solution process.

There are a number of supporting elements which are not discussed here, e.g. a catalogue of developmental laws, a catalogue of typical resolutions of a given contradiction, a huge collection of examples, different supporting methods for different phases of "algorithm" - including a number of tricks for overcoming psychological inertia and barriers, as are well known from "traditional" creativity training.

Conclusion

Although the idea of controlling creativity may at first sound as valid as controlling the movements of stars, one has to accept that it has its points:

- if the problems to be solved belong to the reality of the world,

- if they can be connected with some system which has historical existence,

- if there exist some laws of development which can be studied,

then the primary thing in the supporting of the creativity is certainly the obtaining of a better understanding of the problem area, and any psychological and social properties and hindrances will be of secondary importance. Clearly, the prominent thought of "traditional" creativity support - the producing of more and more wild ideas - begins to look out childish, if not stupid.

4. Discussion.

What is the message of the approach proposed by Altshuller for us in IS research and development? There are at least two viewpoints.

Firstly, in principle it would be possible to proceed according to the same guidelines in IS research - to begin to develop a "theory of innovative information systems". Much of the conceptual apparatus of the Altshuller's theory could be usable as such or with minor changes for the technological component of information systems. The whole of IS is also concerned with social systems and therefore of a more complicated breed, but one could postulate that there exist some general laws guiding the dynamics and development of IS - or better the development of those activities, in which IS are embedded[1]. These could perhaps be extracted by analyzing a number of successful information systems in a manner similar to that used by Altshuller - classifying the systems according to their novelty and impact, recognizing the contradictions and the ways in which they are solved in the "most innovative" systems, etc. The problem is that a huge amount of high-quality data would be needed for such research programme, and there is no "patent information" in the IS area.

In this situation the second viewpoint may be more important. Altshuller's approach should have some influence in the determining the emphasis of our activities and removing some psychological barriers. It tells us that the key to success lies *in the understanding of the nature and dynamics of the system to be changed.* Creativity techniques are unsystematic, blind attempts to overcome the lack of such an understanding.

1 An attempt to postulate one such causality has been made in Kuutti (1989).

It would seem that the term "creativity" - with all those fuzzy and even mystical connotations - can be considered harmful. Very little constructive benefit can be achieved if something is determined as "supporting creativity" or "demanding more creativity", etc. And when we are using it, it draws our attention away from the primary - the phenomenon and its dynamics - towards something secondary, namely tricks and techniques.

I suggest that the term should be rejected and replaced with something more constructive, perhaps "expansive learning" as in Engeström (1987). It is impossible to imagine a situation (in the area of IS research and development) where that replacement would be absurd or would not lead to a more constructive position. All the essentials could be preserved, because expansion - going beyond what is given - is precisely the core of "creativity". There would be only one major difference: unlike "creativity", it is quite difficult to think of learning without some definite substance that is to be learned.

5. References.

Altshuller, G.S. (1988) *Creativity as an Exact Science. The Theory of the Solution of Inventive Problems.* Gordon and Breach Science Publ., New York.

Burrell, G. & Morgan, G. (1979) *Sociological Paradigms and Organizational Analysis.* Heinemann, London.

Engels, F. (1962) *Anti - Dühring. Herr Eugen Dührings revolution in science.* Progress Publ., Moscow. (Translated from 3rd German edition, 1894).

Harré, R. (1981) The positivistic-empiristic approach and its alternative. In (Reason & Rowan 1981), pp. 3-17.

Hirschheim, R. (1985) Information Systems Epistemology: An Historical Perspective. In (Mumford et al. 1985), pp. 13-36.

Klein, H. & Hirschheim, R. (1987) Social Change and the Future of Information Systems Development. In Boland, R. & Hirschheim, R. (eds): *Critical Issues in Information Systems Research*, John Wiley & Sons, Chichester, pp. 275-305.

Klein, H. & Lyytinen, K. (1985) The Poverty of Scientism in Information Systems. In (Mumford et al. 1985), pp. 131-161.

Kuutti, K. (1989) The Impact of Work Development on Information Systems. In *Scandinavian Research in Information Systems*, vol. 1 no 1, pp. 165-176.

MacKinnon, D. W. (1978) *In Search of Human Effectiveness. Identifying and Developing Creativity.* Creative Education Foundation, New York.

Mumford, E. & Hirschheim, R. & Fitzgerald, G & Wood-Harper, A.T. (1985) *Research Methods in Information Systems.* North-Holland, Amsterdam.

Nissen, H.-E. (1985) Acquiring knowledge of information systems - research in a methodological quadmire. In (Mumford et al. 1985), pp. 39-51.

Rantanen, K. (a finnish ASIP-consultant) Private information, January 1989.

Reason, P. & Rowan, J. (eds.) (1981) *Human Inquiry. A Sourcebook of New Paradigm Research.* John Wiley & Sons, Chichester.

Torrance, E. P. (1979) *The Search for Satori and Creativity.* Creative Education Foundation, New York.

Wertheimer, M. (1945) *Productive Thinking.* Harper & Brothers Publ., New York.

Winograd, T. & Flores, F. (1987) *Understanding Computers and Cognition. A New Foundation for Design.* Ablex Publ. New Jersey.

393

Creativity is the answer - but what is the question in system development?

Birthe Lund
Institute for Electronic Systems
Aalborg University Center
9000 Aalborg
Denmark

October 16, 1989

Prepared for
the 12th IRIS Conference on Information Systems Research in Scandinavia.
August 13th - 16th, Skagen, Denmark

Summary

This paper deals with system development as an artistic craft. The system developer is considered a problem solver. It implies that problems cannot be defined objectively, which means taking both the problemsetting *and* the "problem definers" into consideration.
Problem solving is dealing with the world, and it is not *primarily* an analytic skill, which means that it is difficult to elaborate into contextfree rules and general methods. As one realizes that crafts and perspectives depend on the situation, - how it is defined, - the interest of the people involved, - the power relationship, - the stage of the development process, - the current tradition and the work cooperation, one knows why the ideal system developer must be a) an empatic analyser, - b) a creative designer, - c) a good politician and communicator, and d) a clever technician. A, b, and c must be a future part of his/her technique. Without these qualifications, system developers designing special systems will not be able to fulfil special requirements, and the users will choose standard systems instead.
So the paper suggests that the system developer and her/his methods are both open and flexible. If not, the creative elements in problem solving cannot be developed.

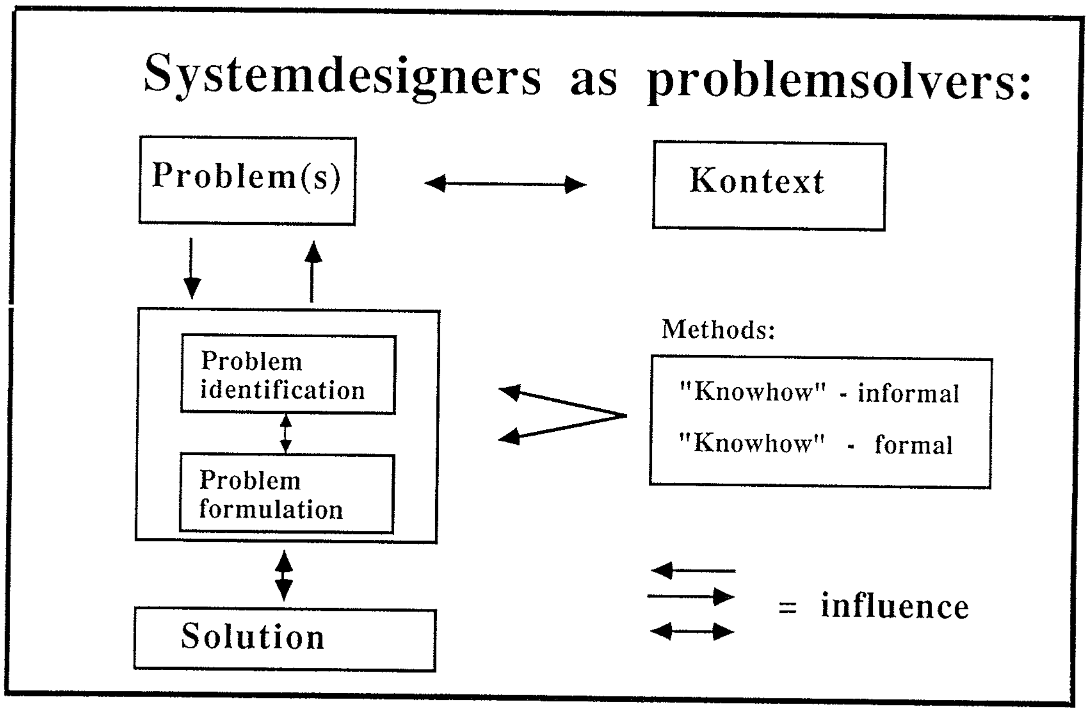
It must be possible to imagine
something other than the fact
of the case, and being able to do so
also means being able to analyze.
But to a lot of people analysis
have become analogous to reductionism.
Isabelle Stengers, kemiker og videnskabsteoretiker,
(Information 29./30. nov. 86)

Introduction

The core theories of system development include hierachical principles of system analysis and functional specifications. It supports well defined and understood tasks. According to empirical observations the greatest difficulties for the system developer concern development goals, process features and structure, and the image field by the participanst about the nature of the development process [Lyytinen 88]. The thinking patterns and approaches (mainly developed in the -70'ies) will not apply any longer, and the IS community may confront a "paradigm" shift. The contents and the problems of system development are in a deep flux and many, if not all, core beliefs of system development will not do in the -90'ies. This suggests that there is a need for a profound rethinking of the whole field - it's theoretical foundations, recommended approaches, management philosophy, and applied techniques, according to Lyytinen's vision of the -90'ies [Lyytinen, 89].
The theme of this conference - system development and creativity - is in itself part of the paradigm shift.

In development of computer-based information systems within the Scandinavian tradition, it has been argued that there is a paradigm shift from a "constructing" to a "process-oriented" view. The process-oriented view results in a focus on what the system developer actually does in the design process, and what kind of skills are needed to fulfil the process. According to this, a skilled system developer must, besides being a clever technician, be a good politician (who can deal with uncertainty and make the right decisions), an intelligent and empatic analyser and finally a creative designer [Andersen, red. 88]. These qualifications become necessary, when one notices that the system developer is both a problemsolver and a coordinator, since system development is teamwork.
The problems, which the system developer must solve, cannot be defined objectively and in advance, which means without taking the problemsetting and the "problem definers" into consideration. One may say that a problem is created by the linguistic acts in which it is identified and categorized [Flores and Winograd]. In this sense "knowing how" to solve a problem and construct a system - seems to be difficult to separate from "knowing that" - theoretical knowing and analysing the problem.

If this "know how" is primarily aquired through action and thereby transformed into experience, such a "know how" will be difficult to elaborate. Based on the Hermeneutic foundation *problem identification* and *problem formulation* is essential and depending on the actual situation, [see figure].



According to this, one may realize that the development processes must be adapted to the actual development context, which means that there is no correct, general way of developing and implementing information systems. Based on the wide experience from various system development processes, methods can however be derived and systemized. These methods must support both problem identification and problemsolving.

While openness and flexibility is essential, when identifying problems, it becomes necessary to systemize when it comes to the constructive elements - problem solving. Current methods seem to support the constructive elements by focusing on a precise and formal procedure, while they do not in the same way enhance innovaty and creativity [Mathiassen 88].
So, contrary to current, established methods, openness and flexibility are essential. Design-methods should not determine a closed sequence of development steps in order to achieve some desired goal - it's primary role is to enhance the *innovation*

and *creativity* of the developers.
If we consider systems development primarily as an art instead of a construction, the designer has to practice an artistic craft as well as an engineering one, but shift emphasis from what is constructed to *how* - the process [Heckel, 84], and then we might be open to a lot of new methods.
Since creativity is a part of a problemsolving process, it would be interesting to figure out what it means to be creative and how to develop creativity. So, in this article, we will concentrate on the fuzzy artistic part.

Art and Creativity

Art and creativity are almost synonymous - to create art simply means to be creative. Creativity, inspiration and ideagenerating are closely combined - without inspiration, no idea will ever develop. As we all know that a certain "input" triggers an idea, it is natural to look for a recipe to select the right "input" - inspiration. Such a recipe seems to be a secret, which special giftet artists keep to themselves - something immaterial. Unfortunately, artists say that only 10% of their work is inspiration, 90% is perspiration. In other words, it is hard work to most artists to create, but this does not change the fact that it is not perspiration, but inspiration we find interesting when we talk about creativity, mainly because we just see the result - the product - and not the whole work process behind it. Nothing comes from doing nothing, though it sometimes seems like the most brilliant ideas suddenly arrive from nowhere, and it is difficult to explain how it happened - Why?
In answering this question, we may look at what creativity means and look at a recipe for "how to develop an idea" from another profession - advertising.
Like the reporter and the advertiser, the system developer is paid to be creative and to develop a product within a limited time, a product which is supposed to satisfy both her/his employer and the readers/users. In this sense both professions are different from what the mythology says about real artists, who are mainly driven by an inner passion and not by ill-gotten gains.
In advertising, it is recognised that creativity is essential. To be labled creative in this business, one must at least have "new ideas". It does not mean to conceive something totally unrelated to anything anyone has ever thought of before.
Creativity simply seems to be a skill consisting of a new way of combining things:

> Alex Osborn: (op. cit. Your Creative Powers, pp.66.) "A creative thinker evolves no new ideas. He actually evolves new combinations of ideas that are already in his mind"[Baldwin.
>
> Ernest Dichter: (op. cit. "Handbook of Consumer Motivation", pp. 411.)

"Creativeness is a basic scientific ability to see the relationship between apparently unrelated things"[Baldwin].

Wilbert S. Ray: (op. cit. "The Experimental Psykology of Original Thinking" pp. 23.) "[the creative thinking process is] the forming of associative elements into new combinations which either meet specified requirements or are useful in some way"[Baldwin].

But how does this new combination of previously unrelated elements - an idea - come about?

Ideageneration

The creative person seems to be a good ideagenerator. The ingrediences of the personified creative advertiser seem to be a connoisseur of the art of living - having an open mind and being able to combine this with intense knowledge of all facets of the subject she/he is advertising for - knowledge and curiosity. It is so difficult to get hold on creativity, simply because it shows in two ways - as a result of a workprocess - a product - which is material, and as an internal thought process - an idea. We will concentrate on the internal process and suggestions for its development in relation to ideageneration. Problem definition is related to ideagenerating - a sudden inspiration can change the perspective and thereby open possibilities for a new solution and solve a problem.
One suggestion for ideageneration is based on traditional analysis.In this you take the different parts of the gathered material, feel them all over, and try to feel the meaning of the different "facts" in order to combine and - later on - synthesize them. Then one must "sleep on it". This is an intermission during which your subconsciousness can work on the problem. When the final idea arises, it then appears as "the only way to do it", and you could kick yourself for having taken so long to figure it out.
The steps of ideageneration can be explained in the following rule of thumbs, - based on Huxley Baldwin's proposal to advertisers -, modified for the system developer:

1. Gather grist for the mill. Accumulate the specific information (about the overall needs the system must fulfil). This should complement your long-range accumulation of general information, earlier experiences and impression of life in general.

2. Grind it all up. Mentally sift and sort, mix and match, what you have learned with what you already know and how you feel. You are looking for that key relationship.

3. Sleep on it. Give your subconsciousness a chance to work on the problems by deliberately getting away from it for a while. (Maybe one should cut off just before knowing how to continue). This is when the trees group themselves into a clearly defined forest.

4. Finally. The solution comes to you, disguised as a flash of sudden inspiration.

5. Look again. Reexamine your idea in the bright light of day. If some of the lustre has work off, polish it, refine it, then test again. If it works, you have yourself an idea.

But the final step, - by James Webb Young called "the cold, gray dawn of the morning after"-, is when we try to figure out, if the little new-born idea is still the marvelous child, it seemed to be, when first you gave birth to it. This part of the ideageneration process is demanding:

> "It requires a deal of patient working over to make most ideas fit the exact conditions, or the practical exigencies, under which they must work. And here is where many good ideas are lost. The idea man, like the inventor, is often not patient enough or practical enough to go through with this adapting part of the process. But it has to be done if you are to put ideas to work in a work - a - day - world. Young: "A Technique for Producing Ideas" s. 52/53 [Baldwin].

We note that the criteria for an idea is related to "functionality" - if it *works*, you have an idea. In other words, the idea must help to solve a problem.
Problem solving can be compared with a jig-saw puzzle. The main problem is to find the pattern, which in real life depends on the *perspective* of the objects. The problem seems to be how to make the pieces of the jig-saw puzzle match. *When* you know the overall pattern, it suddenly becomes very easy. Problem solving is in this sense analytic "brain work".

But even though we know how to specify what a good idea consists of, it would not necessarily be of any help in order to develop one, because expectations seem to play an essential role in producing our ability to make sense out of a potentially infinitely environment.
The experienced system developer does not always know, why certain situations often lead to certain others, but when they do, expectations become associated with the remembered situations, and this will often trigger an idea and lead to a solution.
But it may also produce tunnel vision, - the inability to recognize and adapt to unexpected events [Dreyfus/Dreyfus, 86]. This is the reason why "know how" and "rules of thumbs", developed through practical experience, may create "tunnel

vision"; it is based on intuitive expectations. (But it may also reduce the number of different versions and the time spent on analyze!) In that way, experience may limit the perspective and thereby the number of ways to combine things in new ways - creativity. Central elements in creativity seem to be:

- the ability to analyze and synthezise
- the ability to be aware of and to be attentive to the surroundings.

So openness and flexibility are essential to both system developer and the development methods. On the other hand, useful methods must lead to a certain perspective and thereby help the designer to see the overall picture the jig-saw puzzle.

The creative process

The creative process is a process during which an idea develops into a result. It may include an extension of the 5th stage in the idea - generation process during which one must convince others of one's brilliant ideas. While implementation on the technical level will soon show if the idea works in practice, it is harder to evaluate an idea *as* an idea, because the reexamination is then based, not only on your criteria, but also on those you cooperate with - (let alone one's superiors). If they are not satisfied, you must once again back to "idea generating mode". So, depending on the situation, one must as soon as possible establish a common criteria for the evaluation of the idea - a discurs - as a part of the foundation for design. (Since the result of the evaluation in practice is not totally independent of the power of the decision makers, the bad designer must at least be a good politician!).

In regard to the ideadevelopment steps I want to emphasize, that *reexamining* is a central part of a creative process as a whole, which do not appear clearly. Reexamining is the part which creates more perspiration than inspiration - it is hard work. If one has not refound the overall design - the big idea - in time, the pressure for ideageneration will grow the closer one comes to deadline. Then the subconsciousness does not get a chance to work sufficiently on the problem - you cannot get away mentally - you are forced to be a practical problemsolver - primarily based on impulse - instead of an ideagenerator, who generates the overall plan. So one must admit that planning and timing influence on creativity, and accept that "things take time".

But, on the other hand, we must not forget that inspiration in a way happens on purpose and grows out of dealing with the situation. (That is why we claim that analysing is connected with design). Huntley Baldwin explains a similar phenomenon in this way:

> "The inspiration that comes during this stage of subconscius thinking (which follows a long conscious thinking) is a far cry from just sitting around waiting for a bolt of insight to be hurled down from the gods. This kind of inspiration happens on purpose, and before the deadline" (Baldwin, pp. 27).

What Baldwin expresses is that simply dealing with the problem gives inspiration, and it becomes clear that creativity is related to practical problem solving as well as overall design.

Ideagenerating or problemsolving?

In general, psycological theories about creativity and idea generating has a tendency to deal with the individual and not the coorporation between individuals, they are very general and do not take the contexts in which it is developed into account though creativity shows in both result and process.
As mentioned, ideagenerating in the system developing process is difficult to separate from problemsolving. This becomes very clear, if one takes a hermeneutic tradition, as a new foundation for design.
Within this tradition, knowledge, which is the rawmaterial for inspiration, consists of pre-knowledge established through our being-in-the-world and having to deal with it. In this sense, tools and objects always have a meaning. Things are for ..., or to do with ... By asking - what is this? and at the same time pointing on a hammer, one tends to explain that this is for hammering. In a similar way, we first learn what doors and beds are for, and then the designation of these objects. Things and objects are already interpreted in relation to the background of the viewers. No viewer can escape from this pre-understanding which structures her/his world, because we all are-in-the-world in Heiddeggers terminology.

This is often referred to as "the hermeneutic circle" - based on what we already know, and what we already know comes from being able to understand (Gadamer). By drawing a distinction between existence of an objective world and physical reality and the subjective mental world of individuals, one steps back from the primacy of experience and understanding that operates without reflection, as Heidegger argues.

When we live our lives in a certain way, we take a lot of things for granted. We both bild on and constitute a tradition. It is not until we bring traditions from one culture to another, that we suddenly become aware of our tradition *as* a tradition. In the same way, we suddenly focus on the hammer and the nail - the tool and object instead of the workprocess - hammering - when something breaks down. A break-down creates a problem for a continuing hammering, but it also creates an opportunity to reflect - what happens? What went wrong? What can be done to avoid it? Dealing with the problem creates an opportunity to reflect and/or to solve a problem. (This solution - an idea - makes a new possibility for break-down later on).

In this sense solving problems means creating new possibilities for break-down, and problemsolving can be seen as a part of one big spiral evolution in which we accumulate knowledge (learn) and increase our experience. In this connection, problem solving is not primarily an analytic skill, but a skill developed from dealing with the world - acting.

From this perspective it is difficult to differ between ideagenerating and problemsolving. Problemsolving demands some kind of reflection, which grows out of the situation. Is it the same kind of reflection which generates ideas? Is ideagenerating simply dependent on a problem? I think so, but I think there is an undifined difference between the creative power it takes to solve a more or less given "practical" problem, and the creativity to define a problem and imaginarily visualize something new - according to the overall design.
So designing an edp-system consists of both *problemdefinition* - defining both the problems and which problems the system is to fulfil, and *problemsolving* - how to develop it - imagination. An artistic craft as well as an engineering one. If the developer becomes too much of an engineer she/he tends to give technically perfect solutions to the wrong problem. If she/he acts too artistically, she/he can only partly visualize the possibilities of the system and at worst case, design a system which cannot be constructed.

The creative system developer

To be a creative system developer one must be able to form associative elements into new combinations, which both meet specified requirements and are useful at the same time. The creative person thinks visually and can picture her/his idea, but it is not enough, to fulfil her/his vision into a product - a system, she/he must reexamine, which demands both patience and practical sence.
If the designer thinks too logical and sticks too much to a once given formula or practice, a new perspective will never occur. Tunnel vision will be the result. To imagine new systems one needs a flow of inspiration. Involving the users and thereby their perspective will at the same time give both inspiration and avoid tunnel vision, but it might kill the "free association" by bindings at an early stage in the design process, especially if the system developer is a bad politician or manager, who have not left any space for his own ideas or got the time to go through different ideas or

prototypes.
But to be able to combine things in new ways - not accidentially - one must know the tradition - the craft. We may regard traditional skill as a necessary condition for creativity [Ehn] though this knowledge may create tunnel vision. So, we do not know how to solve these contradictions between the perspective of an "an engineer" and "the one of an artist"; between tradition and innovation, but in order to prevent unquestionable tradition from taking over, one solution is a continuing development of strategies for cooperative design in mixed groups of users and system developers. In other words to be aware of the qualities within different traditions in order to know when an engineering perspective is needed, and how to avoid it, when not.

From one point of view, system development seems to be such a complex work process that a term as system development is no longer sufficient. As one realizes that crafts and perspectives depend on the situation, - how it is defined, - the interest of the people involved, - the power relationship, - the stage of the development process, - the current tradition and the work cooperation, one knows why the ideal system developer must be a) an empatic analyser, - b) a creative designer, - c) a good politician and communicator, and d) a clever technician. A, b, and c must be a future part of his/her technique. Without these qualifications, system developers designing special systems will not be able to fulfil special requirements, and the users will choose standard systems instead.

An alternative is to leave all future system development to a few multy skilled geniuses, who can fulfil the mentioned demands for the ideal system developer and ask them to design a limited number of modules, with an unlimited number of combinations, like the principle in Lego-blocks.
But I do not think any system developer will acknowledge this idea, which could leave the creative combination process directly to the users and leave the average system developer in a consultants role.

References

Andersen, Niels Erik m.fl.: Professionel Systemudvikling Erfaringer, muligheder og handling
Teleteknisk forlag A/S 1988 (1. ed., 1986).

Bækgård, Lars: Edb–systemers kvalitet – empirisk og teoretisk baserede reflektioner
Datalogispeciale, Aalborg Universitetscenter, Mat/Dat, Danmark, 1988.

Dreyfus, Hubert, Dreyfus, Stuart: Mind over Machine The power of Human Intuition and Expertise in the Era of the Computer
Basil Blackwell, 1986.

Ehn, Pelle: Playing the Language–games of Design and Use On skill and Participation
Stencilat Department of Information Science
University of Århus, Denmark.

Habermas, Jurgen: Samtalens fornuft
Rosinante, 1987.

Heckel, Paul: The Elements of Friendly Software Design
Warner Books, 1984 (1. ed. 1982).

Huntley, Baldwin: Creating Effective TV Commercials
Crain Books, 1982.

Lund, Birthe: *Det rationelle paradigmes begrænsning – teknologi og omverdenens forståelse i et nyt perspektiv*
in "Køn og videnskab" - Serie om kvindeforskning nr. 27. AUC forlag 1989.

Lyytinen, Kalle: New Challenges of System Development: a vision of the 90's
Department of Computer Science
University of Jyväskylä, Finland.

Mathiassen, Lars: *Kreativitet og disciplin i system design* in Nordisk Datanytt, nr. 6 1988 - årgang 18.

Ryle, Gilbert: The concept of mind
Penguin Books, 1986 (1. ed. 1949).

Skirbekk, Gunnar: Filosofihistorie Innføring i europeisk filosofihistorie med særleg vekt på politisk filosofi
Universitetsforlaget, 1980 (1. ed. 1972.

Winograd, Terry and Flores, Fernanado: Understanding Computers and Cognition A new Foundation for Design
Ablex Publishing Corporation, 1986.

THE PRINCIPLE OF LIMITED REDUCTION

A Theory of Information System Development

Lars Mathiassen & Jan Stage*

*Institute of Electronic Systems, Aalborg University, Strandvejen 19, DK-9000 Aalborg, Denmark

Abstract

Today, it is conventional wisdom that we can cope effectively with *complexity* in information system development through abstraction and decomposition. The system developers are told to operate in a rational mode, analyzing and describing problems and options and building up *a priori* knowledge on why and how to act.

More recently, much attention has been drawn to the *uncertainties* related to information system development. Iteration, prototyping, and development of successive versions of the same system have been suggested as effective approaches to uncertainty. The system developers are told to operate in an experimental mode, learning from actions and building up *a posteriori* knowledge as a basis for making design decisions.

Both approaches share the somewhat optimistic view that they lead to simpler, more transparent situations that are easier to handle. Both approaches claim that they reduce the capacity and energy needed to perform the task at hand. In practice, however, this is not true: Both approaches do *reduce* the challenge, but what seems to be forgotten is the fact that both approaches at the same time *introduce* other challenges.

This paper presents The Principle of Limited Reduction. In information system development complexity and uncertainty are intrinsically related: Reduction of complexity implies introduction of uncertainty and vice versa. Using a rational approach to cope effectively with the complexity of the product introduces more process uncertainties. And put the other way round: Using an experimental approach to cope effectively with the uncertainties of the product introduces a higher complexity of the process. In practical terms this implies that a rational approach should be supplemented by an experimental management strategy, and correspondingly, that an experimental approach should be supplemented by a rational management strategy.

Keywords

Information system development; complexity and uncertainty; rationality and experimentation; The Principle of Limited Reduction.

1 Introduction

Development of computer based information systems is a challenging and difficult task. Ideally, system developers have to understand and appreciate the details, relations and qualities of the organization to be changed; they have to be professionals in evaluating and handling the technology involved; they must be able to design new systems that meet the requirements of the organization in a way that is both technically and socially implementable; and, finally, they have to operate in a turbulent environment where ideas, preferences, and contractual arrangements change as they perform their task. From this point of view it is not surprising to find that approaches to *complexity* and *uncertainty* has been a major concern within computer and information science.

Today, it is conventional wisdom that we can cope effectively with *complexity* in information system development through abstraction and decomposition (Simon 1969, Dijkstra 1972, Wirth 1976, Parnas & Clements 1985). The assumption is that the system developers are facing too much information in solving a given analysis or design problem. The system developers need to select and structure relevant information and they are told to operate in a rational mode, analyzing and describing problems and options and building up *a priori* knowledge on why and how to act.

More recently, much attention has been drawn to the *uncertainties* related to information system development (Davis 1982, Floyd 1984, Budde *et al.* 1984). Here, the assumption is that the system developers are facing too little information since requirements to a new computer based information system cannot be fully specified in advance. To cope effectively with this they have to develop prototypes, iterate, and develop successive versions of the same system. The system developers are told to operate in an experimental mode, learning from actions and building up *a posteriori* knowledge as a basis for making design decisions.

Both of these approaches claim that they reduce the capacity and energy needed to perform the task at hand. This paper argues against such a naive view and presents The Principle of Limited Reduction. Our claim is that complexity and uncertainty are intrinsically related in information system development: Reduction of complexity implies introduction of uncertainty and vice versa. Using a rational approach to cope effectively with the complexity of the product introduces more process uncertainties. And put the other way around: Using an experimental approach to cope effectively with the uncertainties of the product introduces a higher complexity of the process.

Section 2 contains an exposition of the rational ideal of a priori knowledge, and section 3 contains an exposition of the experimental ideal of aposteriori knowledge. For each approach a typical project situation is described and the basic principles and assumptions underlying it are outlined. Section 4 argues that both approaches share the somewhat optimistic view that they lead to simpler, more transparent situations that are easier to handle. In practice, this is not true: Both approaches do *reduce* the challenge, but what seems to be forgotten is the fact that both approaches at the same time *introduce* other challenges. In practical terms this implies that a rational approach should be supplemented by an experimental management strategy, and correspondingly, that an experimental approach should be supplemented by a rational management strategy. Section 5 contains our conclusions.

2 Knowing: The Ideal of A Priori Knowledge

The ideal of a priori knowledge is mainly concerned with the complexity involved in information system development. The intention is to *reduce complexity* through systematic analysis and extensive use of specifications. The analysis creates clarity by selecting and structuring relevant information. This involves and facilitates rational choice. The resulting information is expressed in *specifications* forming the basis for measurement of progress and division of labour among system developers.

2.1 A Typical Project Situation

The ideal of a priori knowledge is in many situations both relevant and powerful. In the following we will describe one such typical situation.

A software house is to develop a new computer system for a foreign telephone company (MARS 1984, MARS 1985). The software house has previous experience in development of similar systems, and they have succeeded in reaching a contract with the new customer reflecting these experiences. For the first couple of months the project is staffed with a project leader and two half time system developers, and the time is spend on specifying the user oriented functional requirements in detail. The functional specification is accepted by the customer after three months.

The project is now staffed with the project leader an three additional system developers and the situation can be summarized as follows: The contract is settled; a detailed functional specification is accepted by the customer; the task is to design and implement a computer system in accordance with the contract and the specification, and later to deliver this system to the customer. Moreover the system has to be developed within specified time and resource limits without further intensive interaction with the customer. Finally, the key actors in the project have already experience in developing this type of system.

With these characteristics in mind this situation calls for a rational approach. First the system should be designed by relating to the given functional specifications and by utilizing there experiences in designing and implementing similar systems. Then the system should be implemented on the basis of the design specification, and in doing so the project group can be expanded and divided into subgroups each working on their own part of the computer system. Some corrections and iterations would be necessary, but the primary mode of operation is to analyze and think, then to specify, then to split into subsystems that are implemented and tested, and finally to assemble the system.

2.2 Facing Too Much Information

The basic assumptions behind the rational approach can be summarized in the following way (Floyd 1987, Andersen *et al.* 1986, Stage 1989):

- System development is production of computer systems. A computer system developed is delivered to the user organization as a solution to a given problem.
- The structure of the user organization is static. The computer system will be used in different work processes in the user organization. The purpose of and relations between these work processes are unchanged during the development of the computer system.
- The computer system is extrinsically related to the work processes in the user organization. It is possible to separate concerns, and we think and talk about the computer system as something in itself.
- Quality is related to the computer system. Quality can be quantified and measured, and it can only be improved by modifying the computer system itself.
- All relevant properties of the user organization and the computer system to be developed can be specified precisely and unambiguously in a description.

In accordance with these assumptions, the ideal system development project starts with an activity attempting to analyze and understand the context of the computer system. This knowledge forms the basis for extraction of the requirements that are mainly concerned with the functionality of the system. In analysis and design, various kinds of descriptions are the most important media for communicating about the computer system.

In summary, this approach is based on the *ideal of rationality*, i.e. the ideal of knowing why and how before doing. The assumption is that the system developer is facing a complex situation with *too much relevant information.* The system developer is adviced to use abstraction and decomposition to select and specify the information necessary to make decisions and proceed the development effort.

3 Learning: The Ideal of A Posteriori Knowledge

The ideal of a posteriori knowledge is mainly concerned with the uncertainty involved in information system development. The intention is to *reduce uncertainty* through learning based on use of prototypes. *Experiments* provide practical insight into possible solutions, and prototypes serve as effective media for communication and cooperation between system developers and different user groups.

3.1 A Typical Project Situation

The ideal of a posteriori knowledge is, like the ideal of a priori knowledge, both relevant and powerful in many situations. In the following we will describe one such typical situation.

The computer department of a bank is asked to develop a new computer system. The system will provide the customers of the bank with a new type of service, and it will be the first system that will be used directly in the process of selling new services to customers. In case the service is sold, the computer system will typically be used by banking personel while they are having a dialogue with the customer. The computer system will be used in a great number of branches of the bank, and it should be possible to extract various types of information both at the central and local levels.

The project is staffed with people that have extensive experience from other projects in the computer department. They have, however, no experiences with this specific type of system. Moreover, they are to implement the system on a new generation of computer equipment that have recently been installed in the bank.

With these characteristics in mind this situation calls for an experimental approach. Within the bank there is no experience in using or designing systems that serve as direct media between the banking personal and the customers. Up to now the computer department have mainly been engaged in development and modification of basic transaction systems. Facing this new challenge experiments should be designed to learn about possible ways of using computers in a dialogue between a banker and a customer. The problem is not mainly to specify the functionality of the system. The system developers need to know more about the kinds of information, the amount of information, and the ways to structure and sequence information. To learn about this they can design, evaluate, and compare various mock-ups or computer-based prototypes.

3.2 Facing Too Little Information

The basic assumptions behind the experimental approach can be summarized in the following way (Floyd 1987, Andersen *et al.* 1986, Stage 1989):

- System development is production and embedment of various versions of computer-based systems in organizations. A computer-based system includes a technical system and the related human activities. Each new version of a computer-based system creates a new situation and hence new problems in the user organization.
- The structure of the user organization is dynamic. The computer system is tailored to support a specific work process in the user organization. The purpose of this work process and its relations to other work processes are subject to continuos changes.

- The computer system is intrinsically related to work processes in the user organization. We cannot understand the computer system independently of the work processes it supports.
- Quality is related to the use of the computer system. Quality has to be evaluated through personal interpretation, and it can be improved in various ways by changing different elements of the total working situation.
- Relevant properties of the user organization and the computer system being developed can only be described partly by means of various perspectives. The properties are difficult to predict and they are subject to continuos interpretation and negotiation.

In accordance with these assumptions, the ideal system development project includes learning, communication, and negotiation about problems and possibilities as essential activities. These activities are necessary because of the many different and equally relevant kinds of requirements to the system. Mock-ups and prototypes are important media for communicating about the system.

In summary, this approach is based on the *ideal of experimentation*, i.e. the ideal of learning from trying. The assumption is that the system developer is facing an uncertain situation with *too little relevant information*. The system developer is adviced to perform experiments thereby generating and evaluating the information necessary to make decisions and proceed the development effort.

4 Knowing and Learning

The rational and the experimental approach both receive widespread attention. They also share the somewhat optimistic view that they lead to simpler, more transparent situations that are easier to handle. Both approaches claim that they reduce the capacity and energy needed to perform the task at hand. In the following we will argue that this is a naive view not taking into account the *relations* between complexity and uncertainty and the *interdependencies* between a rational and an experimental approach. Both approaches do in fact *reduce* the challenge, but what seems to be forgotten is the fact that both approaches at the same time *introduce* other challenges.

4.1 The Naivity of Simple Reduction

The *rational approach* that has been characterized and illustrated above is based on a simple principle of reduction. This principle can be formulated as follows:

The Rational Claim: A Rational approach based on abstraction and decomposition can effectively reduce the complexity of the task and hence also the capacity and energy needed to perform the task.

If we choose a rational approach we concentrate on processing the information that is there. We analyze technical and functional requirements, we imagine possible solutions, we evaluate them, and based on this we specify a new system. Doing so we assume that:

- the relevant information is available,
- the information is in accordance with reality,
- we are able to select the most useful information as we abstract,
- we have understood the information,
- the specifications are in accordance with the system we are asked to develop,
- the specifications are in accordance with the system we actually develop.

Each of these assumptions express an *uncertainty* related to the rational approach. Using a rational approach we choose to rely on what we know. But what if we learn otherwise from what actually happens through the development process?

The *experimental approach* that has been characterized and illustrated above is also based on a simple principle of reduction. This principle can be formulated as follows:

The Experimental Claim: An experimental approach based on development and use of prototypes can effectively reduce the uncertainty of the task and hence also the capacity and energy needed to perform the task.

If we choose an experimental approach we concentrate on generating and evaluating new information. We set up experiments in which we develop and evaluate prototypes that to some extend resemble or simulate possible solutions. Doing so we assume that:

- the prototypes represent the relevant solutions, i.e. that we do not overlook solutions that will prove better to use and cheaper to develop,
- each prototype represents important properties of a solution,
- we are able to evaluate relevant aspects of the prototype,
- we have the personal strength and the economic and organizational opportunities to throw away a prototype that proved unsatisfactory.

The whole idea of an experimental approach is to produce more information. We try and thereby we hope to come to know more. But how can we make sure that the new information is relevant? Can we be sure that we are able to cope effectively with this information? What seems to be forgotten in the experimental claim is that this approach increases the *complexity* of the development process.

In addition to these specific considerations we can argue in more general terms. Each of the approaches are, as described in sections 2 and 3, based on a series of principles and assumptions, and the validity and usefulness of these are dependent on the characteristics of the situation in which the approach is applied. But each situation is unique with a great many important properties, and in the examples of sections 2.1 and 3.1 we have only described *some* characteristics of a typical project situation. We are only able to clearly relate to one of the two approaches in very few situations. We are dealing with *ideal types*, and in most situations some characteristics point towards an experimental approach while others point towards a rational one. It is seldom a clean cut choice. It is rather a difficult balance.

4.2 The Principle of Limited Reduction

The rational and experimental claims express substantial and useful insight into the development of computer-based information systems. They do, however, represent narrow viewpoints based on an idealized set of assumptions. Each of the approaches express a simple principle of reduction, and if used naively they might easily lead to a situation that *increases* instead of reduces the capacity and energy needed. To avoid this we have to know more about the relation between knowing and learning.

This line of reasoning leads to the following fundamental principle in systems development:

The Principle of Limited Reduction: In information system development complexity and uncertainty are intrinsically related. Reduction of complexity implies introduction of uncertainty and vice versa.

This principle states that the use of a rational approach to cope effectively with the complexity of the product introduces more process uncertainties. And the use of an experimental approach to cope effectively with the uncertainties of the product introduces a higher complexity of the process.

An increased use of methods that impose a formalized behavior on the participants of a system development project is a logical consequence of the rational approach. The basic point is that major problems in system development arise because the work practices applied are too informal. Langefors was one of the first who argued in favour of this and many others have followed this line of argumentation (Langefors 1966).

Preparation and use of formalized descriptions is a common aspect of many of the various attempts to formalize behavior in system development. It is assumed that a precise and unambiguous description is necessary to ensure that the final products meet the specified requirements. Accordingly, most recent methods include a formalized description tool and a set of guidelines for its use.

Many authors have in different ways argued against this approach. Naur argues generally that the problems related to development of high quality software cannot be solved by rules and methods which essentially assume that programmers act like machines for production of programs (Naur 1985). Parnas & Clements are more specifically concerned with system development projects but their basic points support Naur's conclusion (Parnas & Clements 1985). Some of the problems discussed by Parnas & Clements are presented in more detail in section 4.3.

The key point in most objections against the rational approach is that it is generally impossible to predict and describe all relevant aspects and problems in advance. This objection can be rephrased in the following way. The rational approach is an effective way of reducing the complexity that characterize the products of system development. But at the same time it increases the uncertainties related to management of the system development project. From a rational point of view it is impossible to manage a process if all options and products cannot be described in advance. The rational way of coping with product complexity introduces uncertainties that make a rational management strategy irrelevant. Some of these uncertainties are emphasized in section 4.1.

In section 3, the experimental approach was characterized as an effective way of coping with uncertainties related to the products of system development. This idea can also be applied to the management of a system development project. A situation with essential process uncertainties requires that the possibility of introducing new options and revising earlier decisions is left open. This openness towards potential redesign of the development process is the basic characteristic of an experimental management strategy.

The objections against the rational approach has in many cases led to the conclusion that it should be abandoned. In such discussions, the experimental approach is typically presented as an attractive alternative as it is argued that this approach contributes to reduction of some of the major uncertainties related to the products of system development. Through experiments with various prototypes the uncertainties related to the products of system development can be reduced.

The major problem with an experimental approach is that it makes the development process more complex. Every experiment contributes with information concerning a multitude of questions and problems. Unless the purpose of every experiment is made clear in advance, this amount of information may be destructive to the whole project. The information produced combined with presentation of a variety of proposals for design and a large number of participants in the experiments contributes to make the development process very complex.

In section 2, the rational approach was characterized as an effective way of coping with complexity related to the products of system development. This idea can also be applied to the management of a system development project. A situation with essential process complexity requires abstraction and decomposition to select and specify the activities that are necessary to make a certain decision or analyze a specific problem. This decomposition and previous specification of the purpose of each activity in the development process is the basic characteristic of a rational management strategy.

4.3 Practical Advice

The principle of limited reduction implies that a system development project based on a rational production strategy should be supplemented with an experimental management strategy. Correspondingly, an experimental production strategy should be supplemented with a rational management strategy.

Experience shows that the following factors contribute to increase complexity related to the products of system development (Davis 1982, Parnas & Clements 1985, Yourdon 1982):

- The participants are unable to fully comprehend the many details that should be taken into account.
- The system developers have no knowledge about the user organization.
- The basic principles for design of the computer system have been specified in advance.
- The project is based on intensive reuse of existing programs and systems.
- The project is subject to specified time and resource limits.

Experience from practical system development has also shown that the following factors contribute to increase uncertainty related to the products of system development (Davis 1982, Parnas & Clements 1985, Yourdon 1982):

- The users' needs are unclear or ambiguous.
- The participants are unable to specify the requirements.
- The system developers have no experience with the type of computer system they are expected to develop.
- The external conditions of the project are subject to continous change.

In a specific situation, the choice of production strategy may be based on an evaluation of these factors. This production strategy should the be combined with a management strategy as described above.

5 Conclusion

Many discussions concerning system development present the rational and the experimental approach as incompatible strategies. For example, Parnas & Clements (1985) emphasize the rational approach as an ideal we should strive to reach, whereas Floyd (1984) discusses how the ideal denoted as the experimental approach can be realized in practical system development.

In this paper it has been argued and illustrated how a rational and an experimental production strategy contribute to reduce complexity and uncertainty in system development. Furthermore, it has been argued how each of these production strategies require different management strategies. These points has been formulated as a general principle of limited reduction in system development. This principle states that reduction of complexity introduces uncertainty, and vice versa.

The general principle has been combined with reflections concerning its practical implications. In this connection it has been discussed how various factors contribute to increase the complexities and uncertainties that characterize a situation in system development.

References

N. E. Andersen *et al.*, (1986). *Professionel systemudvikling. Erfaringer, muligheder og handling (Professional System Development. Experience, Possibilities, and Action).* Teknisk Forlag.

R. Budde *et al.*, editors, (1984). *Approaches to Prototyping.* Springer-Verlag.

G. B. Davis, (1982). Strategies for information requirement determination. *IBM Systems Journal*, 22(1).

E. Dijkstra, (1972). Notes on structured programming. In O.-J. Dahl *et al.*, editors, *Structured Programming.* Academic Press.

C. Floyd, (1984). A systematic look at prototyping. In R. Budde *et al.*, editors, *Approaches to Prototyping*, Springer-Verlag, Berlin.

C. Floyd, (1987). Outline of a paradigm in software engineering. In G. Bjerknes *et al.*, editors, *Computers and Democracy*, pages 191–212, Gower, Aldershot.

B. Langefors, (1966). *Theoretical Analysis of Information Systems.* Studentlitteratur.

MARS, (1984). Systemudvikling i praksis: Regnecentralen af 1979, århus. MARS-report 3, Computer Science Department, Århus University.

MARS, (1985). Forandring af arbejdsformer i systemudvikling: Regnecentralen af 1979, århus. MARS-report 9, Computer Science Department, Århus University.

P. Naur, (1985). Intuition in software development. In H. Ehrig *et al.*, editors, *Formal Methods and Software Development*, Springer-Verlag, Berlin.

D. L. Parnas and P. C. Clements, (1985). A rational design process: How and why to fake it. In H. Ehrig *et al.*, editors, *Formal Methods and Software Development*, Springer-Verlag, Berlin.

H. Simon, (1969). *The Science of the Artificial.* MIT Press, Cambridge.

J. Stage, (1989). *Mellem tradition og nyskabelse. Analyse og design i systemudvikling (Between Tradition and Transcendence. Analysis and Design in System Development).* Institute for Electronic Systems, Aalborg University.

N. Wirth, (1976). *Algorithms + Data Structures = Programs.* Prentice-Hall.

E. Yourdon, (1982). *Managing the System Life Cycle.* Yourdon Inc., New York.

The Forum Theatre:
Sussie Brandrup and Birgitte Ravn Olesen

CONDITIONS FOR CREATIVITY IN SYSTEM DEVELOPMENT

Andreas Munk-Madsen
Metodica

Berit Thaysen
Crone & Koch edb

Abstract: Creativity is an important issue in system development. We need creativity but we cannot control it directly. In this paper we explore the nature of creativity. We discuss the possibilities of promoting creativity in system development through controlling the conditions that influence creativity. We conclude that we need a culture of cooperation and a professional responsibility to increase creativity in system development. And we suggest that more awareness on the differences in female and male working style could increase creativity.

1. Introduction

Most papers on system development address some tiny part of the field: a specific technique, tool, or activity. The papers and textbooks that do address the totality of system development normally move fast to do some functional decomposition. Then they also treat tiny parts of system development. This kind of approach tends to neglect overall features that permeate all system development activities. Creativity is one of these features.

We wish to take the opportunity offered by the theme of the 12th IRIS conference to explore the issue of creativity in system development. The basis for this exploration is our mutual experience in practice, research, and education. We have worked more than 20 years with system development. We have experienced a great number of system development processes as participants, consultants, and observers. Thus we find it relevant to present our view of system development practice although we are not able to give formal evidence to our claims. However, we think it would be an interesting research project to examine the viewpoints put forward in this paper.

The purpose of the paper is to present some preliminary conclusions regarding the nature of creativity and the possibilities of promoting creativity. In section 2 we discuss why the issue of creativity is so important in system development. In section 3 we define creativity as "richness of useful ideas". Based on the ideas of selected authors we discuss the nature of creativity. Section 4 is an attempt to classify the various conditions for creativity. In section 5 we evaluate current system development practice in relation to this classification. Section 6 concludes the paper by proposing some principles for improving creativity in system development.

2. The Importance of Creativity

Why is creativity an important issue in system development right now? The fundamental reason is that the nature of system development is development work. This reason is augmented by the particular conditions for system development as compared to other kinds of development work. Finally the current status of the field of system development in the historical social development poses a demand for extra creativity. We will elaborate these reasons.

System development is development work.

The very word development implies that new things must be conceived. Thus a richness of useful ideas is relevant.

Development also means that existing problems, conditions, and tools must be matched by the new solution. Thus not all new ideas are applicable. This means the ideas must be useful.

Furthermore the problems, conditions, and tools to be matched are moving while we develop the solutions. In order to hit this moving target we must work with a variety of solutions. This also demands a richness of ideas.

The particular conditions of system development increase the demand for creativity.

The rapid development in hardware entails that system development often involves the use of new tools. Thus there is a particular scarcity of ready-made solutions in our field. This can only be compensated by richness of useful ideas.

System development is an integrated technical and organizational activity aiming at a moving target. This entails a high degree of uncertainty. The rapid increase in the number of system developers entails that there is a low degree of system development experience in the organizations. Uncertainty coupled with inexperience result in frequent crises. In a situation of crisis a richness of useful ideas is in demand.

The status of system development in the historical social development accentuates the demand for creativity.

The society is undergoing a rapid development, that is characterized by demands for higher efficiency - particularly in administration and service industry. These demands can only be met through organizational changes involving decentralization and higher skills. This development in turn poses requests for new edp-based systems that provide a variety of informations instead of just performing a set of predefined routine functions. Creating new edp-based systems requires a richness of ideas.

Furthermore a changed approach in system development is needed when we are not just automatizing existing functions. We can not reach a solution that support a new kind of business entirely through analysis when there is no old system to analyze. Instead we need a richness of useful ideas.

3. The Nature of Creativity

The importance of creativity in system development entails that we should discuss how we promote creativity. To do this we must discuss the nature of creativity. In this section we discuss what creativity is and how it works. We base this discussion on ideas from a variety of fields: Philosophy, advertising, consulting, and management. We have selected these ideas because they give us insight into the nature of creativity.

A dictionary (Gyldendal 87, our translation) defines creativity as:

> "richness of ideas and the ability to implement the ideas."

We subscribe to this definition as we find it covers most use of the word. However, we want to stress that the first part of the definition is most important and we also want a shorter definition. Therefore we define creativity as:

> "richness of useful ideas."

We will give a few supplementary comments on this definition. Although the concept of "idea" has connotations of newness, creativity does not imply originality. The ideas should not necessarily be new, only they should hitherto not have been present on the debate in the situation. We define an idea to be useful if it improves a process in relation to the purpose of that process.

Creativity is an ability that every person has to some extent. We talk about individual and collective creativity and we make no sharp distinction between these phenomena in this paper.

Let us now discuss the nature of creativity. What we are searching is a richness of useful ideas. Where do they come from? Mao Zedong (Mao 63) says:

> "Where do correct ideas come from? Do they drop from the skies? No. Are they innate in the mind? No. They come from social practice, and from it alone; they come from three kinds of social practice, the struggle for production, the class struggle and scientific experiment. It is man's social being that determines his thinking."

Ideas are related to practice. This is the materialistic foundation of creativity. What we want to study is the detailed interplay between practice and our minds when we generate ideas. The field of advertising has a long experience in creativity. An experienced practitioner in this field, David Ogilvy (Ogilvy 83) says:

> "I am supposed to be one of the more fertile inventors of big ideas, but in my long career as a copywriter I have not had more than 20, if that. Big ideas come from the unconscious. This is true in art, in science and in advertising. But your unconscious has to be **well informed**, or your idea will be irrelevant. Stuff your conscious mind with information, then unhook your rational thought process. You can help this process by going for a long walk, or taking a hot bath, or drinking half a pint of claret. Suddenly, if the telephone line from your unconscious is open, a big idea wells up within you."

The consequence of what Ogilvy says here is that we cannot consciously control our creativity, since our unconscious is involved. What he proposes is to improve the conditions for creativity. This involves two guidelines. We must be well informed. And we must listen to our unconscious.

Gerald Weinberg has written a book on consulting (Weinberg 85). He also says that the unconscious plays an important role in developing ideas. Concerning how to listen to our unconscious, he says:

> "Certainly, it would be a waste of time to develop your unconscious if you don't have one, but the evidence indicates that most people do. But, by it's nature it tends to be hidden from you unless you practice looking for it. Yours may not express itself to you by songs, like mine does, but perhaps it communicates through slips of the tongue, gestures, one-lines, puns, catchwords, flashes of mental pictures, body posture, inexplicable noticing of objects, mistaking one person for another, or a combination of several such phenomena."

In his book on project management Meilir Page-Jones (Page-Jones 85) comments on creativity when he talks of brainstorming meetings:

> "There is but a fine line between creativity and insanity. What stanches creative ideas is the mind's own critical demon that prematurely dubs them to be impossible. The purpose of brainstorming is to loosen the flow of ideas - both good and bad - by deliberately silencing our carping internal critic."

To be creative we must be open to ideas that are not good enough yet. We must be in a not too criticising state of mind. Page-Jones continues:

> "One guideline for a brainstorming meeting is to think laterally - that is, to make leaps of creati-

vity, rather than follow a logical train of thought. A second guideline is to build on the ideas of others, rather than to attempt to find flaws in those ideas. Following these guidelines produces a stream of unconventional proposals. Although most of these proposals will turn out to be insane, one or two may prove to be both original and workable."

This introduces a new guideline: Listening to others. In his book on management Gerald Weinberg (Weinberg 86) has further systematized some guidelines for creating new ideas, which he calls error, theft and copulation. "Errors" means that mistake sometimes lead to useful ideas. If only we are open to these ideas. "Theft" means that we can get many ideas by studying the work of others. "Copulation" means that we may combine two ideas into one that is better than the two original ideas.

We can sum up the workings of creativity as follows. We cannot control creativity directly. But we can improve the chances for creativity by following some guidelines.

These guidelines say that we must be well-informed about practice, we must be open to irrational ideas and to our own unconscious, we must be experienced in creative search patterns such as the ones Weinberg calls theft, errors, and copulation, and we should know how to work together to stimulate creativity for example at brainstorming meetings.

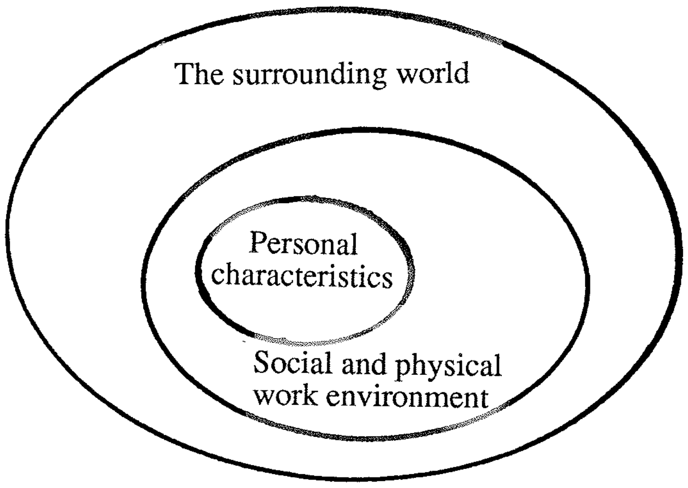
In the next section we take a systematic look at conditions under which this behaviour may thrive.

4. Conditions for Creativity

In section 3 we argued that creativity cannot be controlled directly, but indirectly we can influence creativity by controlling its conditions. In this section we will discuss the conditions that influence creativity.

Creativity is influenced by many conditions involving our personal characteristics, our social and physical work environment, and the surrounding world.

There is a mutual interplay between these conditions. The social and physical work environment influence our daily work. It sets the frame under which we may feel comfortable and develop our personal characteristics. On the other hand, each person influences his or her social and physical work environment. Similarly the surrounding world limits our personal characteristics and our work environment. And the surrounding world is influenced by us and our work environment.



Conditions for System Developer Creativity.

In section 3 we concluded that creativity involved being well-informed and listening to our unconscious. Therefore the personal characteristic of highest importance for creativity are knowledge, values, and motivation.

Knowledge is acquired through education and reflection on experience. More education and experience normally means more knowledge.

Knowledge is not enough. To be creative we must also be willing to use our knowledge and we must be able to do so. Our inclination to use our knowledge is determined by the motivation offered by our work situation. We must believe that creativity is important. It is important to feel that others are interested in our ideas and use them. Our ability to use our knowledge is influenced by our experience in listening to our unconscious.

The social and physical work environment are the close surroundings in our daily work situation.

The most important social environment is our project group and our department. It is especially the project culture and the department culture that influence creativity. Creativity thrive in cultures that value exchange and development of knowledge. Such cultures offer challenges to the persons involved and reward the persons who accept the challenges.

We must be able to steal ideas. This we can do in discussions with colleagues and through reading literature. Creativity grows when you are able to use other people's knowledge to inform your unconscious.

Our creativity is influenced by our ability to concentrate on our tasks. We must have stillness around us. We should not have too many urgent problems, and we should not work on too many different tasks at the same time. We must have time to gather information, to think, and to let the thoughts mature.

Our physical work environment also influence our creativity. We need reasonable conditions regarding light, noise, temperature, air, room, and tools. It is also important that we can influence the style of these physical surroundings according to our own taste.

The surrounding world influences our personal characteristics and our work environment. It consists of the organization in which we work, its corporate culture, our professional network, and the rest of the society.

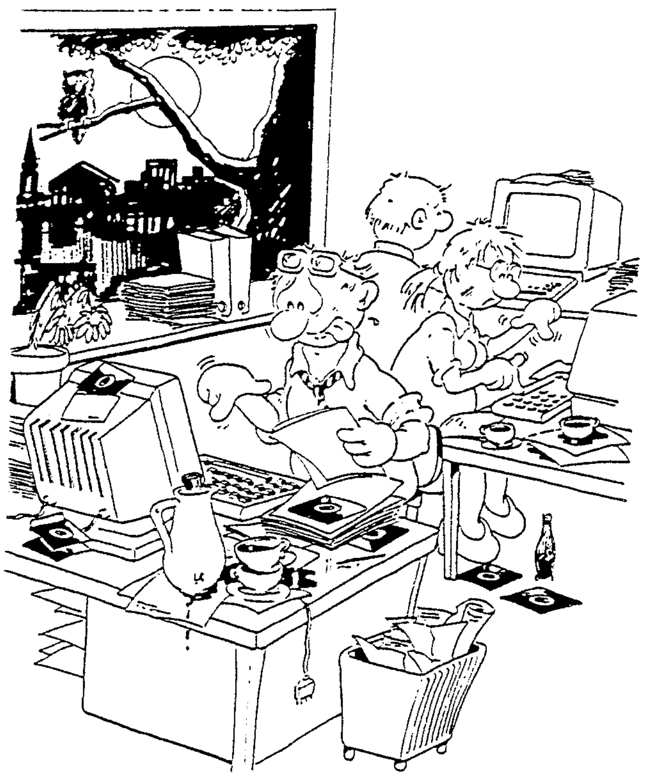
The rules and the culture of the organization influence the way we behave in our daily work situation and our possibility to change the conditions for our creativity. Our professional network is an important source for inspiration and information. It offers possibilities to exchange experiences with system developers from other companies.

5. The Typical Work Situation of the System Developer

So far we have discussed idealized conditions for creativity. Now we want to discuss the typical work situation for a system developer. It is interesting to ask: How are the actual conditions? Is it possible to be creative in system development?

When we evaluate the typical work situation of the system developer we find big differences from one situation to another and that give us some difficulties in presenting general conclusions. What we will do is to state what we find in many cases and present that as a trend.

In the typical work situation of the system developer we find many possibilities for creative behaviour and many examples of creativity. We also find a number of factors that restrict creativity and examples of uncontrolled creativity, which lead to never ending development.



The majority of the system developers have a good education. Many have several years of formal computer science education. There is also a tradition for more supplementary education than in most other fields. Related to creativity this means that the necessary basic knowledge normally is available.

We find differences in the motivation offered by the work situation, and this means that there is differences in the ability to act creatively. Sometimes the ability and the willingness to use the available knowledge creatively is restricted by the system developers' own unprofessional values regarding working habits: overly ambitious designs are not uncommon, impossible schedules are accepted, working overtime is necessary - and may even be considered fun, division of labour means isolated work.

In many cases we also find that system developers have a narrow imagination regarding their own possibilities for changing the situation. This means that the determination to create better conditions for creativity is limited by the belief that this is of no use.

Almost everywhere the work situation is characterized as an informal environment with relaxed and cozy social conventions that stimulate the creative behaviour. A strong project culture is also a tradition. The culture is used as a means to improve the well-being of the system developers. But the culture is seldom regarded as something that can be changed and used to support creative behaviour. Maybe because we do not learn anything about cultures and changing cultures in our education.

We also find stressed social work environment often characterized by late schedules, poor project economy, dissatisfied users, insufficient project management, and the system developers' bad values regarding working habits. This stressful situations means that it is difficult to plan your own work. It is difficult to get time to gather inspiration from others, to listen to your unconscious, and to let ideas mature.

Project organization is a well established tradition. It is implemented as a division of labour combined with relative autonomy regarding methods. In reality system developers are responsible for their own working practices, and this means that they have many possibilities to create favorable conditions for cooperation in a creative manner. However, they are normally inexperienced in close cooperation and the division of labour is often implemented as a way to delegate tasks for isolated solutions. Therefore there is only a limited use of the ideas of others.

System developers are considered as technicians - uninterested in their physical surroundings. We often find system developers messed together in small rooms interrupted by printer noise and telephone conversations. Creativity can hardly thrive under such conditions.

The increasing demands for quality and efficiency influence the work situation of the system developers. Sometimes these demands arise because the professional capabilities are oversold to the customers. This entails a constant demand for useful ideas and insufficient time to develop the ideas. This could be seen as a challenge motivating creative behaviour, but it could also be seen as a stress factor restricting creative behaviour.

System development managers often come from an engineering tradition. They think system development projects may be run and controlled as construction work. They do not understand that creative processes needs another sort of control. In fact they do not trust the requirements for creativity. This entails that many projects are out of control. The system developers do not get the necessary calm to develop useful ideas.

Supplementary education is a tradition in the field. There are many courses available, although most are tool-oriented and not people-oriented.

It is a small field with strong professional networks. There is some tradition for exchange of experience between different organizations, and this means many possibilities to gather inspiration to act creative.

Returning to the question in the beginning of this section we conclude that there are many possibilities for creative behaviour, but there is a limited tradition for using these possibilities.

Thus we find that traditions - more than rules and formal methods - limit the creativity in system development.

6. How Do We Promote Creativity in System Development?

In the previous sections we say that many conditions influence creativity. We think there are huge differences between system developers regarding which conditions are the most important. We also think there are huge differences between organizations regarding the status of these conditions. For these reasons we hesitate to recommend any specific actions that should be generally applicable.

However, we think that strategies for increased creativity in general should focus on the relation between the individual system developer and the environment. This relation is important because creativity is connected to something inside the system developer that we can not influence directly. What we can influence are the conditions in the environment.

The general strategy that we propose consists of two parts: A culture of cooperation and increased professional responsibility. It can be summed up as a simultaneously female and male working practice.

A culture of cooperation

We get many of our ideas from others. We need the encouragement of others. And we need others to test, criticize, and elaborate our ideas.

Therefore creative cooperation is important and must become a tradition. It should be a natural part of the culture that we just do these things. So we may concentrate on the ideas.

It is difficult to change cultures. So we must pay attention to the culture whenever we can: In the professional debate, at project establishment meetings, in education, and in recruitment.

A professional responsibility

Each system developer is responsible for her or his professional creativity. This is the natural consequence of the fact that creativity is something inside the individual person.

What can we do externally to increase this responsibility? We can discuss creativity. We can - as users, consumers, citizens, managers, and colleagues - demand higher product quality and process efficiency. And we can argue that there should be adequate resources and conditions.

Female and male working practices

Creative behaviour is the result of many individual factors. Some of these factors may be grouped into what we call female and male working practices. We use the words "female" and "male" deliberately as we do not want to say that all women are like this and all men like that. We think both men and women have female and male working practices. But women in general are more characterized by female working practices and men by male working practices.

Female working practices are characterized by lack of self consciousness. There is a need for encouragement and support. It is important that the environment is well-organized, familiar, and comfortable. Listening to the ideas and needs of others is a natural pattern of behaviour. Female working practices fit into a culture of cooperation.

Male working practices are characterized by competition. Cooperation may be restricted by a tendency to individual performance. New, untested, and challenging tasks are preferred. Male working practices conform to professional responsibility.

Our conclusion is thus that simultaneously female and male working practices lead to higher creativity. But in business as well as in education male working practices have a higher ranking than female working practices. Competition seems to be valued more than cooperation, and this attitude can hinder awareness of the valuable in the female working practices and influence the willingness to combine the two sorts of working practices.

References

Gyldendal 87	"Gyldendals Fremmedordbog", 10th edition, Gyldendal, København 1987.
Mao 63	Mao Zedong "Where Do Correct Ideas Come From?", 1963. Quoted from "Five Essays on Philosophy", Foreign Languages Press, Beijing, 1977.
Ogilvy 83	David Ogilvy: "Ogilvy on Advertising", Pan Books, London, 1983.
Page-Jones 85	Meilir Page-Jones: "Practical Project Management", Dorset House Publishing, New York, 1985.
Weinberg 85	Gerald Weinberg: "The secrets of consulting", Dorset House Publishing, New York, 1985.
Weinberg 86	Gerald Weinberg: "Becoming a Technical Leader", Dorset House Publishing, New York, 1986.

A Survey of Approaches in ISD Methodology

Peter Axel Nielsen
Department of Mathematics and Computer Science
Institute of Electronic Systems, Aalborg University
Strandvejen 19, DK-9000 Aalborg, Denmark
+45 98 13 87 88 - pan@iesd.auc.dk

Abstract

A vast number of very different methods for developing computer-based information systems have been proposed by researcher and practitioner over the last two decades. They form a motley spectrum and they apply to different parts of the development process and to different modes of thinking and acting.

On top of these methods we find approaches for answering the fundamental question: Which methods in which situations? This paper seeks to survey these approaches by mapping each of them unto a model of a generic approach. This enables an evaluation of the approaches according to how well the question is dealt with.

It is argued that though it is important to characterise and compare methods and relate them to specific features of situations in ISD it is necessary to move from approaches that assess methods and situations in general to approaches that facilitate a process where the assessments are made in a specific situation.

1. Introduction

What to do and how to do it have always been a crucial element of information systems development (ISD). The "whats" and "hows" of ISD are often refered to as *methods*, i.e. sets of coherent guidelines that informs practitioners and when interpreted give rise to concrete thinking and acting. Many methods have been proposed during the last two decades, e.g. Structured Analysis/Structured Design, Jackson Systems Development, Structured Analysis and Design Techniques, ISAC[1], to name a few. Methods are very different in nature as they rely on different assumptions about information systems and the process of development and they relate to different parts of the process.

When a practitioner is faced with a situation in information systems development he or she will as part of that have to know about different methods. There are several good reasons for this. Firstly, a variety of methods already exists and it is not clear in the outset what distinguish them. Secondly, a number of failures in ISD can be traced back to the application of methods inappropriate or inadequate in the situation at hand (amongst other things). Thirdly, if the ideal is multi-perspectivation,[2] where a variety of different perspectives are applied it becomes important to understand what aspects of a specific situation focus should be on and thus how different perspectives of methods support enquiry into these aspects.

1 DeMarco (1974), Jackson (1983), Gane & Sarson (1977), and Lundeberg *et al.* (1979).
2 cf. Nygaard & Sørgaard (1987).

Much research in ISD is *about* methods and in that sense within the field of ISD methodology. (It is not difficult to get the impression that more effort is going into talking about methods than into actually developing and improving methods). Different approaches have been proposed during the last years in this field. Some of these approaches deal, some way or another, with the fundamental question: Which methods in which situations? It is these approaches that will be addressed in this paper. The purpose of the paper is to show to what extent and in what way the question have been dealt with so far. The question is intentionally somewhat pragmatic in order to find out about the potential practicality of the approaches. It may well be that the researchers behind the approaches did not have such a pragmatic question in mind when they came up with the approach. Nevertheless, I choose to view them as dealing with it to some extent.

Each of the approaches will be mapped unto the model in Figure 1. The model is in my view a generic approach in ISD methodology made up of the activities derived from the basic pragmatic question of which methods in which situations. In systems terms[3], the model is a systems model conforming with the following systems definition.

Systems definition:
A system for ISD researchers and practitioners to decide on which methods to seek to use in a specific situation based on knowledge about methods and ISD.

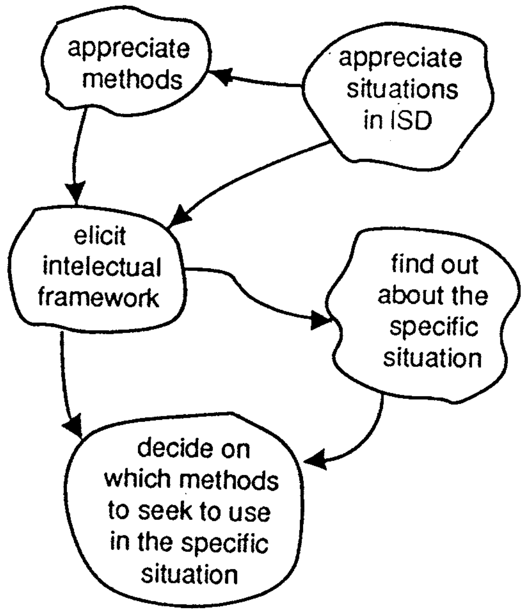

appreciate methods
appreciate situations in ISD
elicit intelectual framework
find out about the specific situation
decide on which methods to seek to use in the specific situation

Figure 1: A systems model for approaches in ISD methodology

The mapping of an approach will be done by asking the following questions for each activity in the systems model:

- If it is already done as part of developing the approach:
  - How was it done?
  - What was achieved?

3 cf. Checkland (1981) and Wilson (1984).

- If it is not done already:
    - ° How should it be done?
    - ° What is the desired outcome?

Section 2 is about the approaches dealing with methods and situations and especially their relation. While Section 2 describes the approaches, evaluates some of their features and discuss how they answer the fundamental question, Section 3 contains a discussion of the reasons for going even further than the evaluated approaches and focus on the process of answering the basic question in a specific situation rather than answering it once and for all. The concluding remarks are given in Section 4.

2. Approaching Methods and Situations

The approaches can be separated into four by two distinctions. Firstly, there is the distinction between whether the approaches are based on assertions generalised from practice or on theoretical assertions. This is not to say that the practice-based assertions does not rely on theoretical considerations and vice versa, but only that what have been the prime concern in the research that led to the approaches. Secondly, there is the distinction between whether the approaches associate with methods solely or with both methods and situations in ISD and their relation. Approaches concerned with methods alone focus on the differences between methods assuming that situations in ISD are alike, while approaches concerned with the both methods and situations assumes that properties of different situations can be related to different features of methods.

2.1. Theory-based on Methods

Taggart and Tharp have studied a number of methods and asserted to what extent each of the considered methods treats the four aspects[4]:

Development process: Treatment of evaluation in analysis phase?

Information: Recognition of information key characteristics? Treatment of information-needs scope? Degree of sohpistication?

Decision making: Recognition of decision process? Awareness on decision-making hierarchy? Awareness on varying degrees of ability of human information processors?

Organisation: Recognition of organisation environment? Discussion of organisation subsystems? Recognition that information vary with management function and level?

Each of the considered methods are assessed on a scale from 1 to 3, where 3 means "significant treatment of aspect". The TRACE method[5], for example, is rated: 3, 2, 2, 2, 1, 1, 1, 2, and 2 in the above aspects.

Mapping Taggart and Tharp's approach unto the generic approach in Figure 1 reveals the following. The methods have been appreciated by reading the literature in

[4] Taggart & Tharp (1977), p. 275. Twentytwo methods are considered, though some of them may today more rightly be regarded as general theories rather than methods, e.g. Langefors (1963).

[5] Altman *et al.* (1971).

which they are presented. The understanding of the methods is therefore detached from how they actually can be practiced. The framework, i.e. the knowledge about which aspects of methods are important, is then elicited on the basis of this theoretical methodological understanding and a pre-understanding of ISD without reference to practice. The appreciation of situations has not been done and in this approach it would not be meaningful, because according to the framework all situations are alike and calls, for example, for equal 'treatment of evaluation in analysis phase'. The two remaining activities are for the same reason meaningless in this approach. The question of finding an approapriate method for evaluation in analysis phase can in this framework be answered once and for all without taking any notice of the specific needs of the situation at hand.

By studying the literature on various methods Wood-Harper and Fitzgerald classifies the whole spectrum of methods into six categories and establish thereby a taxonomy. They distinguish methods by paradigm, model, and objective and find that the six categories are[6]:

General systems theory methods: are based on a paradigm where reality is a system and by manipulating general and abstract systems models and thereby understand the situation as a systems obeying general rules. It is worth noticing that the authors does not identify any successful methods in this category.

Human activity systems methods: Reality is considered problematic and the situation is explored by various (soft) systems models and this hopefully leads to some improvement of the situation. Checkland's Soft Systems Methodology is one such approach.[7]

Participative methods: are based on systems thinking and seek to establish satisfying work for people. Methods in the socio-technical tradition are within this category.[8]

Traditional methods: view reality as a stable data process, i.e. as a function relating input to output which is produced by a computer system specified in a formal logic.

Data analysis methods: Reality is a stable data structure and information systems are developed by identifying and formalising data objects.

Structured systems methods: are based on the reductionistic assumption that reality is flow of information and information systems are developed by analysing this flow in a formal and hierarchical way. SASD is one such method.[9]

If we map this approach unto the generic approach we find that it is significantly different from Taggart & Tharp's. Though the methods have been appreciated by reading about them Wood-Harper & Fitzgerald take the methods as a starting point rather than ISD. Without seeking to appreciate situations in ISD an intelectual framework is elicited to show important differences between methods. The core of the approach is these differences explained in terms of paradigms (ontological and epistemological assumptions), models preferred, and objectives. Turning to the two activities about specific situations show that in this approach they are not done and there is not guidelines included on how to do them. Hence, while the framework asserts that methods are

6 Wood-Harper & Fitzgerald (1982). Wood-Harper (1988) uses the same categories now validated by action research.

7 Checkland (1981). Note, that I refer to SSM as a method dispite Checkland is naming it a methodology.

8 Wood-Harper and Fitzgerald refers to Mumford *et al.* (1978) and Mumford (1979). Today, it is evident that many methods falls into this category, see, for example, Bjerknes *et al.* (1987).

9 DeMarco (1979), and so is Gane & Sarson (1977).

different it is assumed that knowing about the differences will enable the actors in a particular situation to decide on which methods to seek to use. Later, Wood-Harper, Antill & Avison have set up the Multiview approach based on this framework as it is basically a statement about in what sequence each category should be active.[10]

Wasserman, Freeman and Porcella have examined 24 methods by a questionnaire focussing on overview rather than details of the methods in order to see to what extent they can be integrated with the programming language Ada.[11] In their paper they discuss:

Life cycle coverage: Do each of the methods cover: requirements analysis, functional specification, design, implementation, validation, and evolution? Taking JSD[12] as an example shows that it covers all phases but validation and evolution.

Applicability: Are the methods suited for: embedded, science and engineering, operating, tools, data processing and data base, and/or expert systems? Are the methods suited for small, medium, and large sized systems? JSD, for example, is satisfactory suited for science and engineering systems, insufficient experience with expert systems, and well suited for all other systems of all sizes.

Technical concepts supported: To what extent do each of the methods support: function hierarchy decomposition, data hierarchy decomposition, interface definitions, data flow, sequential control flow, concurrency and parallelism, formal program verification? For example, JSD does not cover funtion decomposition and formal verification.

Workproducts and representation: What are the prescribed workproducts and the representation schemes used? Some of the workproducts in JSD are: entity and action lists, entity structures, etc., and some of the representations used are: tree structures, data flow diagrams, etc.

Quality assurance: What quality assurance techniques are applied to workproducts? How is the final system validated against the original requirements? The quality assurance in JSD is done by author/reader cycles, structured walkthroughs, and inspections and the validation is done by manually checking the transformation of specification to implementation.

Usage: What is the degree of difficulty in using each of the methods? How many projects and organisations have used the methods? JSD is moderately easy to use and has been used in between 2 and 10 projects in between 2 and 10 organisations.

Viewed from the generic approach this approach is of the same kind as the two previous, but the research engaged is very different. Wasserman *et al.* discuss at length the theoretical reasons for and their understanding of ISD as a basis for a design of a questionaire about methods. The only source for appreciating the methods is the questionaire answered by those who developed each of the methods investigated. To the extent that the developers can provide useful evaluative information about their own work this gives some insight into the methods. The intelectual framework they elicit consists of the aspects described together with an assessment in all aspects of all methods as exemplified with JSD. The approach is mainly a comparison of methods with respect to some features of ISD (appreciated theoretically), but these features do not distinguish between situations in ISD and the two activities to do with a specific situation are therefore not considered in this approach at all.

---

10 Wood-Harper *et al.* (1985).

11 Wasserman *et al.* (1983). The most well-known of these 24 methods are: ISAC, SADT, JSD, and SASD; 9 of the 13 methods in Olle *et al.* (1982) are compared here, see later.

12 Jackson (1983).

The two approaches of Bergland and Davis are not considered in this paper as they relate to methods from the restricted perspective of programming.[13]

2.2. Practice-based on Methods

The CRIS conferences (Comparative Review of Information Systems Design Methodologies) is a sizable effort in comparing methods.[14] The first conference was established around a test case concerned with organising a conference (The IFIP Case) and each contributor was asked to apply a method to the test case. Each contribution was reviewed by a committee according to a large set of questions about the methods in general and how well they handled the test case. The purpose was that of taking stock and present a spectrum of methods.

The second conference had the purpose of feature analysis of the methods presented at the first conference. A number approaches for assessing the features came up based on very different theoretical backgrounds, of which only a few will be described in detail in the following.

Brandt discusses and compares all the methods from CRIS 1.[15] The discussion and comparison is made by means of a taxonomy:

Origin and experince: The environment the method is developed in and the experience with its use.

Development process: The phases that the method cover, especially the easiness of obtaining programs. ISAC,[16] for example, covers analysis fully and design to some extent, but with unclear connection to programs.

Data model: The concepts, formalisation, and abstraction the systems models is based on. ISAC is not based on any particular data model.

Iteration and tests: The procedures for tests, validation and verification of products. In ISAC verfication is not considered

Representation means: The language being it graphical, formal, based on forms, and especially whether the language is a canidate for automation. All results are presented in tables and graphs (being composed of a small number of simple symbols) is the full assessment of ISAC.

---

13 Bergland (1981): Four methods, functional decomposition, data-flow design, data-structure design, and programming calculus, are compared with respect to code-level (e.g. abstraction, communication, clarity, control flow), module-level, system-level. Davis (1988): A number of methods are compared in order to reduce the ambiguity, inconsistency, and incompleteness of software specifications.

14 Olle *et al.* (1982), (1983), and (1986). Basically, the CRIS effort is based on practice and concerned with methods without relating to different situations, although it is a collection of somewhat different approaches. The exception is Wasserman *et al.* (1983) which is not based on practice, not even the experince gained in CRIS 1. The 13 methods in CRIS 1 are: ACM/PCM by Brodie and Silva, CIAM by Gustavsson, Karlsson, and Bubenko Jr., D2S2 by Macdonald and Palmer, DADES by Olivé, EDM by Rzevski, Trafford, and Wells, IML by Richter and Durchholz, ISAC by Lundeberg, ISSM by Sølvberg, NIAM by Verheijen and Bekkum, REMORA by Rolland and Richard, SDLA by Knuth, Hálasz, and Radó, SYSDOC by Aschim and Mostue, and USE by Wasserman.

15 Brandt (1983).

16 Lundeberg (1982), according to Brandt (1983).

Documentation: The integration of and stress put on documentation. Large volumes of documentation are generated using ISAC.

User orientation: The knowledge that must be possessed in order to use the method and ways of participation. The objective of ISAC is to enable the users do the development themselves.

Tools and automation prospects: Do automated tools exist or are they planned to be developed. For example, no automated support is reported for ISAC.

Based on assessment of each of the methods in CRIS 1 a superficial comparison is made.

Olivé describes the similarities between methods.[17] The methods in CRIS 1 are compared with respect to:

Levels of abstraction: There are five levels of abstraction related to information systems: external (describing the context for the information system), conceptual (describing the possible states of the context an information systems must respond to), logical (describing the operational aspects of an information system), architectural (describing the overall architecture of the information system), and physical (describing in detail the previous level). Taking ISAC[18] as an example shows some of the assessment: it covers all level but the conceptual.

Types of information system: There are basically two types of information system: data base systems and general systems. To each of these types more detailed aspects can be assessed. ISAC, for example, leads to general systems.

Bodart *et al.* evaluates methods by relating them to three cycles of information systems development: abstraction, decision, and control.[19] Falkenberg *et al.* assess and compare four methods by interpretation of concepts and they argue that the best aspects of each of the methods can be syntisised into a 'best' method.[20] Iivari and Kerola analyse the features of the methods by relating them to a cybernetic framework they call the PIOCO model of ISD from which they ask 85 questions about each method.[21] Kung compares three methods in a time perspective on information modeling.[22]

All the approaches in CRIS 2 can be mapped unto the generic approach in very much the same way. All the approaches described appreciate the relevant methods by studying the outcome of an application of the methods to the IFIP Case made by the developers. They do not apply the methods themselves but rely on what the developers of the methods achieve and it is assumed that important features of methods are revealed by this common case that works as an representative for all situations in ISD. The main differences between the approaches stem from different appreciations of ISD. They all asume that situations in ISD are more or less alike or that the similarities are the most important. The purpose of CRIS 2 is comparative review of methods and all the approaches compare methods according to how they view ISD. Again we find that the activities

17 Olivé (1983).

18 Lundeberg (1982), according to Olivé (1983).

19 Bodart *et al.* (1983).

20 Falkenberg *et al.* (1983).

21 Iivari & Kerola (1983). Eventhough the framework is based on contingency assumptions they argue that methods should be flexible not that different situations ask for different methods; in this sense they view situations as different but within the scope of a method, hence the classification of their approach in this section.

22 Kung (1983).

about specific situations are not dealt with because of the fundamental assumption that situations are alike that underpins the whole idea of CRIS.

It is interesting to notice that eventhough all the approaches in CRIS 2 set out to assess the important features of methods very few of the features they find are the same. This suggests that it is possible to produce an endless stream of features and distinctions to evaluate and compare methods against.

Floyd has made a study of the methods: SASD, JSD, SADT.[23] First, each of the methods are assessed on their own terms, e.g. "the notion of 'action' is confusing since it has no time dimension" is one such experience. Then a few concepts for categorisation of methods are offered:

Area of application: is characterisedby, e.g. system size, application orientation, e.g.

Perspective: The views and values embodied in or implied by the method.

Guidelines: Basically consisting of tools, techniques, and principles of organisation.

Theory: The theory it is based on with respect to what a computer-based information system ought to be viewed as and with respect to what information systems development is.

Coherence: The extent to which the guidelines are related to each other in a convincing manner, e.g. by a overall strategy.

Coverage: The extent to which support is given to the parts of ISD considered relevant by the method.

Product-orientedfeatures: i.e. features of the product of ISD. The features are studied in three aspects: universe of discourse (including, e.g. objectives, requirements, funtions), development stages, and problem areas (such as man-mashine interface, data organisation).

Mapping this unto the generic model shows first of all that it is based on a somewhat different relationship with practice. The appreciation of methods is done by "teaching courses in which the methods were presented, tried out by students on a case study and subsequently evaluated."[24] The appreciation of situations in ISD seems to be done implicitly by adhering to a view on ISD. The framework is elicited from the experience gained from the students' practice and based on a theoretical discussion of how to categorise methods. The framework is intentionally vague in the direction of categorising and setting a taxonomy for methods as the purpose is more to clarify the area of methods as no suitable criteria for a taxonomy can be found, Floyd argues. As to the two remaining activities no insight is offered.

2.3. Theory-based on Methods and Situations

Ciborra, Bracchi and Maggiolini are most likely the first to seek to relate the differences of ISD methods to different situations of ISD.[25] They define a framework based on a Simonean understanding of problem solving where the process of matching methods and situation is seen as a *search* in a task environment. The task environment consists of: the nature of the organisation and information system, the available computer tech-

[23] Floyd (1984) and (1986).
[24] Floyd (1986), p. 20.
[25] Ciborra *et al.* (1980).

nology, the relation between users and analysts, and project management. It is stated that "different task environments imply different problem spaces and, above all, different methods and approaches"[26] and by assessing the factors it is possible find which methods are appropriate in a particular situation. The factors are assessed by the following criteria:

Organisation and Information System: Is there a unknown and complex relationship or a known programmed relationship between the organisation and the information system?

Technology: Is the technology new or is it conventional data processing?

The Users and Analysts: Are the analysts dependant on involvement with the users or can they work detached? Do the users act passively or actively?

Project Management: Is it a low risk or high risk project?

Eleven methods have been assessed in this framework in order to show how different methods match different task environments. E.g. ISAC[27] is found to be able to cope with: both known and unknown relationships between organisation and information system, conventional data processing, participation, a medium risk project.

Mapping this approach unto the generic approach divulges that situations in ISD are appreciated theoretically by relating ISD to the Simonean interpretation of problem solving and other theoretical contributions to the understanding of the specific nature of ISD. Methods are appreciated by Ciborra *et al.* by studying the literature on each of the considered methods. The intelectual framework is then elicited by arguing the outlook of specific problem solving of ISD based on their understanding of situations, but without reference to their knowledge about methods. As an allustration of the framework a number of methods are assessed by the criteria offered in it. In a specific situation the framework may be used as a means for finding out which methods are appropriate in that situation (i.e. task envorinment). By assessing the specific situation according to the above criteria in the framework it is then possible to to decide on which methods to seek to use simply by finding out which methods available are able to cope with the characteristics of the task environment.

Davis' approach for matching methods and situations based on uncertainty determination is probably the best known in its field.[28] Based on a Simonean understanding of human information processing Davis finds that there are four strategies in ISD spanning a spectrum of uncertainty they can cope with: asking, deriving from an existing system, traditional analysis, and experimentation.

1. *Define characteristics:* Four elements in ISD affect uncertainty: organisational context (.e.g. reduce uncertainty: stable and well-defined organisation), information system, users and analysts (e.g. increase uncertainty: little prior training or experience with similar information system).

2. *Evaluate process uncertainty:* The characteristics affect the process uncertainty: uncertainty with respect to existence and stability of useful requirements (e.g. arise from lack of well-understood model of organisational context), uncertainty with respect to users' ability to specify requirements (e.g. arise from changes in the use of information), uncertainty with respect to ability of analysts to elicit requirements.

[26] Ciborra *et al.* (1980), p. 52.
[27] Lundeberg *et al.* (1978).
[28] Davis (1982).

3. *Evaluate overall uncertainty:* Evaluate the three process uncertainties to arrive at an overall level of process uncertainty.

4. *Select a primary strategy:* Based on the level of uncertainty a primary strategy and one or more methods are selected. 'Asking' can cope with low uncertainty and 'experimentation' with high uncertainty.

Davis' approach seems to be based on a theoretical understanding of situations in ISD primarily based on Simon's work on human information processing and Davis' own understanding of ISD. Methods are appreciated mainly by studying the literature on methods, though there are exceptions where the appreciation is based on experience from practice[29]. From this, a framework is elicited where the main concern is the level of uncertainty, which must be assessed in order to find appropriate methods. The framework also consists of an assessment of the level of uncertainty each of the considered methods can cope with, for example, that ISAC is a method of traditional analysis and can cope with a medium-high level of uncertainty. In Davis' approach we find for the first time a thorough explanation of the two activities to do with a specific situation. The first three stages described above is how the finding out of the situation should be done where the final outcome is an assessment of the overall uncertainty of the situation as a whole. Anyone who have tried to use Davis' approach will have noticed that this is not a straight forward activity because the detailed uncertainties are difficult to evaluate and believe in. The decision on primary strategy then forms the remaining activity. The strength of this approach seems to be in the thinking about relevant aspects and by that finding arguments for a decision on which methods to seek to use.

Lately, Benyon and Skidmore have based on the work of Wood-Harper & Fitzgerald and CRIS proposed 'the analysts tool kit' consisting of five complementary methods[30]: soft systems, participative, traditional, structured systems, and data centred methods. They argue that the analyst should be skilled in all of these methods and choose the most appropriate for the situation at hand based on a set of characteristics:

- The reason for using the method (exploration, communication, experimentation, or prediction).
- The level of detail desired.
- The management style of the organisation.
- Organisational size, arrangements, hierarchy and norms.
- The nature of the systems trigger.

This approach maps unto the generic approach in very much the same way as we saw with the approach of Wood-Harper and Fitzgerald. Additionally, they have formed a few aspects to consider when finding out about the specific situation, i.e. the three last aspects in the above, and a few aspects to be used in the decision on which methods to use, i.e. the two first aspects.

2.4. Practice-based on Methods and Situations

Episkopou and Wood-Harper have proposed a framework for choosing appropriate methods.[31] They take as their starting point Davis' approach and Wood-Harper and

[29] Munro & Davis (1977).
[30] Benyon & Skidmore (1987).
[31] Episkopou & Wood-Harper (1986).

Fitzgerald's taxonomy from which they argue their "process of choosing". The process is characterised by five elements, of which only the last depends on the others:

Assess Problem Owner: Identification of the problem owners, i.e. the persons that perceive a problem in the situation. Thereafter, the personality/cognitive style, the skills, background and experience, and the ability to specify requirements are assessed.

Assess the Problem Content System: The problem content system is the system that contains the problem and its environment. The following may be assessed: history of the environment, resources (man, computers, money, time), size and maturity, interest groups, company characteristics, uncertainty.

Assess the Problem Solver: This is a kind of self-analysis with respect to: the personality/cognitive style, the skills, background and experience, and the ability to elicit requirements and evaluate their correctness and completeness.

Assess Methods: The characteristics that Episkopou and Wood-Harper feel are important are: the ideology of the method, the tools associated, the links with Churchman's inquiry systems, the cost and time span.

Formulate the Problem Solving System: This is basically a process of choice where different characteristics are matched. The hints are scarce on this element. However, some characteristics are already matched: available time with time required, available man resources with man resources required, cognitive style of PO and PS with ideology of approach and link with inquiry system, skills and experiences of PO and PS with tools.

According to Episkopou and Wood-Harper this approach is based on more than 60 action reserach projects with different methods, and in this sense the appreciation of methods and situations is absed on generalisations from practice. The framework is elicited as a composition of all these aspects (the five categories above). It seems that the direct relation to practice ends here as the action research have dealt with methods and not the approach itself. Hence, when it comes to the two activities about a specific situation the framework suggests how to do them in some detail, but no evidence on the soundness or usefulness of this part of the framework is given. The finding out about the specific situation is done by assessing the problem owner, the problem content system, and the problem solver. The decision on methods is done by formulating the problem solving system by choosing methods that match the characteristics of the situation.

NIMSAD (Normative Information Model-based Systems Analysis and Design) is an approach developed and used by Jayaratna.[32] Basically, it is a framework for understanding and evaluating methods and their use in ISD. The framework consists of eight stages or more rightly of eight necessary activities in ISD:

1. Introduction to the "real world": The stage where a segment of the real world as a dynamic situation is introduced in terms of a network of structures, tasks, people, technology, processes, resources, facilities and the relationships between involved actors are established.

2. Understanding the situation of concern: The stage where the analysts attempts to understand the aspects of the situation concerning the clients. It is argued that the analysts should be alert to a wide range of values, assumptions and capabilities, e.g. structuring processes (including methods), experiences, etc., especially the analysts' own.

32 Jayaratna (1986).

3. *Diagnosis:* The stage is concerned with explicitly stating an image of the present situation. The image should be expressed in two forms: the logical arrangements of roles, structures, flows, processes, etc. and the physical/practical arrangements of the elements in the former.

4. *Prognosis outline:* The stage of defining the problem owners' expectations as more than accepting the expectations as part of the situational conditions.

5. *Systems analysis:* The conceptual mapping of the prognosis on the diagnosis. i.e. to find the areas of improvement and for systems design and debating them as Notional Systems and from this select one Notional System.

6. *Logical Design:* Taking the Notional System as given this stage is about describing what is necessary to support the Notional System.

7. *Physical Design:* Through this stage "a physical design model is generated for a given set of physical constraints/resources."

8. *Implementation:* This stage is concerned with the construction of the Notional System.

Jayaratna sees his approach "as an aid to the understanding of systems analysis and design in general, as a meta-level framework for evaluating methodologies and as an aid to the evaluation of methodological practice."[33] NIMSAD has been developed and used over a period of years and by this experience is gained with methods, situations, and NIMSAD itself. The appreciation of methods and situations seems to a large extent to be based on such experience, and the framework is elicited as a generalisation of experience gained through practice in action research projects. Though, the framework has been used to evaluate methods generally it is significantly different from the appraoches described above in the sense that it is much more a thinking and re-thinking the use of different methods while being in the midst of a project. The two remaining activities about a specific situation utilises this framework in guiding systems practice.

3. From General Assessment To Situational Thinking

Schön has made a thorough analysis of how practitioners think and act.[34] Schön distinguishes between two different modes of thinking: technical rationality and reflection-in-action.

Technical rationality is seen as instrumental problem solving. The practitioner takes a goal as given and by selecting the best means seeks to reach this goal.[35] Hence, technical rationality is based on the assumption that there is consensus about the goal and that it is visible and clear. The selection of best means is done by applying the relevant scientific theory.[36] Knowledge is mainly a result of science and can be seperated from practice, where professional knowledge is seen as three parts: an underlying basic science, an applied science, and skills and attitudes. In this sense practice and science (research) remain seperate where the knowledge produced by research is specialised and standardised theories and techniques that can be applied by practitioners to diagnose

33 Jayaratna (1986), p. 86.

34 Schön (1983). The book is about the thinking and acting of professional practitioners in general. It can, however, easily be argued that information systems development can be understood as professional practice, too; empirical findings shows that Schön's framework can serve as a profound interpretation of the thinking of ISD practitioners, cf. the diaries in Jepsen & Nielsen (1986).

35 Schön (1983), p. 21ff.

36 Schön (1983), p. 34.

situations and solve problems and practical knowledge is knowledge about the relationship of means to ends.

Viewing ISD methodology from the view-point of technical rationality implies that situations are rigorously classified into a set of categories with associated methods that can be used in that situation. Given a particular situation it is therefore possible in a scientific way to determine what method would be appropriate in that situation.

In contrast to technical rationality, reflection-in-action is seen as problem setting where each situation is considered unique: "the practitioner approaches the practice problem as a unique case ... seeks to discover the particular features of his problematic situation, and from their gradual discovery, design an intervention."[37] The three most significant differences between technical rationality and reflection-in-action are:

	Technical rationality	**Reflection-in-action**
Situations:	Falls into scientifically defined categories.	Are unique, complex, uncertain, and value-conflictual.
Knowledge:	Is a result of science and must be seperated from practice.	Is inseparable from action.
Practice:	Is fundamentally different from research (science); practice is application of theory and research is production of theory.	Includes research.

Table 1. Significant difference between the two view-points.

Schön discusses at length why technical rationality must be abandoned and why we must see professional practice as reflection-in-action. Firstly, a situation is fundamentally unique:

> "Even when a problem has been constructed, it may escape the categories of applied science because it presents itself as unique or unstable. In order to solve a problem by the application of existing theory or technique, a practitioner must be able to map those categories onto features of the practice situation. ... But a unique case falls outside the categories of applied theory; an unstable situation slips out from under them."[38]

Secondly, taking a goal as given ignores problem setting. Schön argues that in practice problems are not given, they are constructed from the problematic situation. The process of setting the problem is:

> "a process in which, interactively, we *name* the things to which we will attend and *frame* the context in which we will attend to them."[39]

That is, that ends are not given, ends and means are mutually dependant and is therefore set in the same process.

There are four aspects of reflection-in-action that to a large extent characterise it: *problem definition*, *experiments* with actions and what they lead to the most fundamental experimental question being "What if?", a *repertoire* of examples, images, understandings, and actions to bring past experience into the situation by "Seeing as", and *stance* towards enquiry as an actor rather than an observer.[40]

Viewing ISD methodology from the point of view of reflection-in-action reveals the process of matching situations and methods quite different from technical rationality and what we have seen in the previous section.

The experiments that the practitioner conducts in order to find a way out of the problematic situation is founded in a set of pragmatic and subjective value-judgements.

[37] Schön (1983), p. 129.
[38] Schön (1983), p. 41.
[39] Schön (1983), p. 40, Schön's italic.
[40] Schön (1983). p. 128-164.

In this case, finding out which methods to seek to use calls for experiments with several methods and they seem to be based on the following questions[41]:

- Can we use this method in this situation?
- Do we like what we get when we apply this method?
- Have we made the situation coherent?
- Have we made it congruent with our fundamental values and theories?
- Have we kept enquiry moving?

This suggests that practitioners are far more likely to choose a method they like and appreciate than an "optimal" method for the situation. Furthermore, the repertoire consists of much more than a number of ISD methods as the repertoire encompass the experience of the practitioner with aspects of ISD as well as with methods.

Let us now try to conclude this discussion of approaches in ISD methodology by stating that *we must rely on both technical rationality and reflection-in-action.*

On the one hand, it is possible and useful to investigate the features of methods and situations. By this knowledge about methods and the situations where they are potentially useful is build up. If, for example, experience from practice shows that DeMarco's data-flow technique is not useful when describing issue-based work in a bank, because it cannot in logic or in practice be routinised, then this experience is generalisable into knowledge about data-flow detached from the situation where it was experienced. This kind of knowledge is prominent within and belongs to the domain of technical rationality. The whole previous section showed several approaches based on technical rationality, and they contribute to ISD methodology in different but limited ways. The contribution from this domain is characterised by *general assessment* where the obtained knowledge (assessments) exist detached form a particular situation.

On the other hand, the general argument put forward by Schön does apply to ISD methodology, too. This suggests that we must strive at practicing reflection-in-action in the process of matching methods and situations. The work ahead lies in finding out how to to do the *situational thinking*, i.e. that part of the process where we cannot rely on the general assessments because the uniqueness of the situation demands for assessments and arguments specific to the situation.

What should be done then? Firstly, we must also seek to empirically understand the process in which the situational thinking takes place. One of the most common drawbacks of the approaches described here is that they at most report experiences with methods and situations and not the process of answering the question of which methods in which situations.[42] Secondly, we must find out how to facilitate this process of answering the question rather than trying to answer it once and for all.

4. Concluding Remarks

The approaches have been presented and discussed in one of four groups as showed in the following table.

[41] Schön (1983), p. 133. The questions are applied to the specific elements of problematic situations concerned with choosing methods.

	Theory-based	**Practice-based**
Methods:	Taggart & Tharp 1977 Wood-Harper & Fitzgerald 1982 Wasserman *et al.* 1983	Olle *et al.* 1982 Olle *et al.* 1983: Brandt 1983 Olivé 1983 (Bodart *et al.* 1983) (Falkenberg *et al.* 1983) (Iivari & Kerola 1983) (Kung 1983) Floyd 1984 and 1986
Methods and situations:	Ciborra *et al.* 1980 Davis 1982 Benyon & Skidmore 1987	Episkopou & Wood-Harper 1986 Jayaratna 1986

Table 2. The approaches categorised into the four groups.

By mapping the different approaches unto the generic approach it was showed how each of them answered the fundamental question of which methods in which situations.

The approaches we have seen so far are based on, what Schön will call, technical rationality. We can in many respects, though, utilise the knowledge embedded in these approaches, but we will have to do more than relying on the general assessments. We will have to reflect-in-action (in Schön's terms) and actually have to think, too. The general assessments are part of this situational thinking and to go even further than this a deeper understanding of the epistemology of situational thinking about methods is envisioned.

References

Altman, J. W., A. W. Leavitt, S. C. Shannon & S. T. Hovly, (1971). *Handbook of Methods for Information System Analysts and Designers.* National Technical Information Service Publication, Springfield.

Andersen, N. E., F. Kensing, M. Lassen, L. Mathiassen, A. Munk-Madsen & P. Sørgaard, (1989). *Professional Systems Development.* Prentice-Hall, New York. In print.

Bjerknes, G., P. Ehn & M. Kyng, editors, (1987). *Computers and Democracy.* Gower, Aldershot.

Benyon, D. & S. Skidmore, (1987). Towards a tool kit for the systems analyst. *The Computer Journal,* 30(1):2-7.

Bergland, G. D., (1981). A guided tour of program design methodologies. *Computer,* (October):13-37.

Bodart, F., A. Flory, M. Leonard, A. Rochfeld, C. Rolland & H. Tardieu, (1983). Evaluation of CRIS 1 IS development methods using a three cycle framework. In Olle *et al.* (1983).

Brandt, I., (1983). A comparative study of information systems design methodologies. In Olle *et al.* (1983).

Checkland, P. B., (1981). *Systems Thinking, Systems Practice.* John Wiley, Chichester.

Ciborra, C., G. Bracchi & P. Maggiolini, (1980). A multiple-contengency review of systems analysis methods and models. In H. Lucas, F. Land, Lincoln & Supper, editors, *The Information Systems Environment,* pages 47-60, North-Holland, Amsterdam.

Davis, A. M., (1988). A comparison of techniques for the specification of external systems behavior. *Comm. ACM,* 31(9):1098-1113.

Davis, G. B., (1982). Strategies for information requirements determination. *IBM Systems Journal,* 21(1):4-30.

DeMarco, T., (1974). *Systems Analysis, Systems Design.* Yourdon Press, New York.

Episkopou, E. & A. T. Wood-Harper, (1986). Towards a framework to choose appropriate IS approach. *The Computer Journal,* 29(3):222-228.

Falkenberg, E., G. M. Nijssen, A. Adams, L. Bradley, P. Bugeia, A. L. Cambell, M. Carkeet, G. Lehmann & A. Shoesmith, (1983). Feature analysis of ACM/PCM, CIAM, ISAC and NIAM. In Olle *et al.* (1983).

Firth, R., B. Wood, R. Pethla, I. Roberts, V. Mosley & T. Dolce, (1987). *A Classification Scheme for Software Development Methods.* Research Report CMU/SEI-87-TR-41, ESD-TR-87-204, Real-Time Methodologies Project, Software Engineering Insitute, Carnegie Mellon University, Pittsburgh, Pennsylvania 15213.

Floyd, C., (1984). *A Comparative Evaluation of System Development Methods.* Seminar paper, Technische Universität Berlin, Institut für Angewandte Informatik. Seminar at Dansk Datamatik Center, Denmark.

Floyd, C., (1986). A comparative evaluation of systems development methods. In Olle *et al.* (1986).

Gane, C. & T. Sarson, (1977). *Structured Systems Analysis: Tools & Techniques.* Improved System Technology Inc., New York.

Iivari, J., (1987). Assessing IS design methodologies as methods of IS assessment. In Bjørn-Andersen, N. & G. B. Davis, editors, *Information Systems Assessment,* North-Holland, Amsterdam.

Iivari, J. & P. Kerola, (1983). A sociocybernetic framework for the feature analysis of information systems design methodologies. In Olle *et al.* (1983).

Jackson, M., (1983). *Information Systems Development.* Prentice-Hall, London.

Jayaratna, N., (1986). Normative information model-based analysis and design (NIMSAD): A framework for understanding and evaluating methodologies. *Journal of Applied Systems Analysis,* 13:73-88.

Jepsen, L. O. & P. A. Nielsen, (1986). *Dagbogsskrivning til styring i systemudvikling, (in danish, engl: Diaries for Effective Management in Information Systems Development).* Masters Thesis, Dept. of Computer Science, Århus University.

Kung, C. H., (1983). An analysis of three conceptual models with time perspective. In Olle *et al.* (1983).

Landry, M. & J.-L. Malovin, (1983). The mirage of universal methods in systems development. *Journal of Applied Systems Analysis,* 10.

Langerfors, (1963).

Lundeberg, M., (1979). A systematic approach to information systems development, I: Introduction, II: Problem and data oriented methodology. *Information Systems, 4.*

Lundeberg, M., (1982). The ISAC approach to specification of information systems and its application to the organization of an IFIP Working Conference. In Olle *et al.* (1982).

Miles, R., (1985). Computer systems analysis: The constraints of the hard systems paradigm. *Journal of Applied Systems Analysis,* 12:55-65.

Munro, M. C. & G. B. Davis, (1977). Determining management information needs - a comparison of methods. *MIS Quarterly,* 1(2).

Mumford, E. & D. Henshall, (1979). *A Participative Approach to Computer Systems Design.* Halsted Press, New York.

Mumford, E., F. Land & Hawgood, (1978). A participative approach to computer systems. *Impact of Science on Society,* 28(3):235-253.

Nygaard, K. & P. Sørgaard, (1987). The perspective concept in informatics. In Bjerknes, G., P. Ehn & M. Kyng, editors, *Computers and Democracy,* pages 374-394, Gower, Aldershot.

Olivé, A., (1983). Analysis of conceptual and logical models in information systems design methodologies. In Olle *et al.* (1983).

Olle, T. W., H. G. Sol & A. A. Verrijn-Stuart, editors, (1982). *Information Systems Design Methodologies: A Comparative Review.*

Olle, T. W., H. G. Sol & C. J. Tully, editors, (1983). *Information Systems Design Methodologies: A Feature Analysis.* North-Holland, Amsterdam.

Olle, T. W., H. G. Sol & A. A. Verrijn-Stuart, editors, (1986). *Information Systems Design Methodologies: Improving the Practice.* North-Holland, Amsterdam.

Saarinen, T., (1988). System development methodology and project success: An empirical assessment of contingency models. In Kaasbøll, J., editor, *Report of The 11th Information Systems Research Seminar in Scandinavia.,* IRIS XI, pages 515-535, University of Oslo.

Sage, P., B. Galling & A. Lagomasino, (1983). Methodologies for determination of information requirements for decision support systems. *Large Scale Systems,* 5:131-157.

Schön, D., (1983). *The Reflective Practitioner.* Basic Books, New York.

Shomenta, J., G. Kamp, B. Hanson & B. Simpson, (1983). The application worksheet: An evaluation tool for matching new development methods with appropriate applications. *MIS Quarterly,* pages 1-10.

Sibley, L. J. B., (1986). The evolution of approaches to information systems design methodology. In Olle *et al.* (1986).

Taggart, W., Jr. & M. O. Tharp, (1977). A survey of information requirements analysis techniques. *Computing Survey,* 9(4):273-290.

Tranfield, D., (1983). Management Information Systems: An Exploration of Core Philosophies. *Journal of Applied Systems Analysis,* 10.

Wasserman, A. I., P. Freeman, and M. Porcella, (1983). Characteristics of software development methodologies. In Olle *et al.* (1983).

Wood-Harper, A. T., (1985). Research methods in information systems: Using action research. In E. Mumford *et al.*, editors, *Research Methods in Information Systems*, pages 169-191, North-Holland, Amsterdam.

Wood-Harper, A. T., (1988). *A Comparison of Information Systems Definition Methodologies*. Presentation at the Systems Conference at University of Lancaster, July.

Wood-Harper, A. T., L. Antill & D. E. Avison, (1985). *Information Systems Definition: The Multiview Approach*, Blackwell, Oxford.

Wood-Harper, A. T. & G. Fitzgerald, (1982). A taxonomy of current approaches to systems analysis. *The Computer Journal*, 25(1): 12-16.

Yadav, S. B., R. R. Bravoco, A. T. Chatfield & T. M. Rajkumar, (1988). Comparison of analysis techniques for information requirement determination. *Comm. ACM*, 31(9):1090-1097.

Designing for Creativity — Toward a Theoretical Basis for the Design of Interactive Information Systems

Kenneth Nilsson

Institute of Information Processing
Administrative Data Processing
University of Umeå
S90187 Umeå
Sweden
krn@cs.umu.se

Abstract

Interactive information systems aim at realizing an environment that admits support for the user to work in a tentative way in his or her activities in the early stages of a problem solving process. Tasks that call for such a support can be design tasks, planning and allocation problems or tasks that are difficult to handle analytically. This kind of information systems should allow for the user to take advantage of his own creative ability. For the design of interactive information systems we need a theoretical basis that gives us a better understanding of human activities in which computers can be used and the involvement of computers in these activities.

A theoretical approach in this direction has been developed by Winograd and Flores. This paper presents some of their main ideas and their criticism of what they call the rationalistic tradition which is predominant within computer science. Their approach is contrasted in the paper by treating certain aspects of the epistemology of the French philosopher Bachelard, that indicate limitations of the approach of Winograd and Flores. There is an interesting difference between Winograd and Flores on one hand and Bachelard on the other concerning the role attributed to reflection and formalization versus pre-understanding and unreflected action in developing new knowledge. This difference indicates that certain aspects of creativity do not fit into the framework of Winograd and Flores. Some consequences of this observation and their relevance for a continued research on a theoretical basis for interactive information systems are discussed.

Keywords Creativity, design, interactivity, interactive information systems, problem solving, cognition, human activity, representation

0. Introduction

In this paper we will discuss a theoretical background for the design of interactive information systems. Information systems of this kind are aiming at realizing an environment that admits a support for the user to work in a tentative way in his or her activities in the early stages of a problem solving process. There is a number of tasks that call for such an approach, e.g. design tasks, planning and allocation problems and tasks intractable to analytically handle (Section 1).

A fundamental question in this context concerns the theoretical basis on which to found design principles for interactive information systems. We claim that a theoretical basis for better understanding human activities, in which computers can be used, and the involvement of computers in these activities is necessary (Section 2). The work of Terry Winograd and Fernando Flores[1] represents a research in this spirit and an account of some of their views is given in Section 3.

Interactive information systems should allow for the user to take advantage of his own creative ability. The contribution of Winograd and Flores is discussed with respect to creativity (Section 4). As a perspective contrasting that of Winograd and Flores, we present in Section 5 the epistemology of the French philosopher Gaston Bachelard. There is an interesting difference between Winograd and Flores on one hand and Bachelard on the other concerning the role attributed to reflection and formalization versus pre-understanding and unreflective action in developing new knowledge. The paper is completed by a discussion and some conclusions on this difference.

[1] Winograd, Terry and Flores, Fernando (1986), *Understanding Computers and Cognition — A New Foundation for Design*, Ablex Corporation, Norwood NJ.

1. Interactive information systems

Computers have often been seen as a means to automate human tasks. This has certainly been the case within government and business administration, where a number of routine data processing tasks have been automated. Traditional methods of systems development refer to applications giving information support to strongly formalized and stable work processes, e.g. order handling or inventory bookkeeping. Also, in developing advanced computer applications the striving to construct programs, that solve ever more advanced tasks automatically, is prevalent. Most computer applications of this kind can be seen as having the course of computations quite predetermined and from a problem solving perspective giving support in a late stage of the problem solving process. In a man-machine context this means that the role of the user in the computing process is on the whole reduced to input the correct data at the correct time as seen from the computer program's view.

However, there is a number of tasks which are intractable to analytically handle and for which heuristic approaches are motivated. Often these tasks may be characterized as "unstructured". They may be described as tasks where solution objectives are ambiguous, numerous or not operational, where the process required to achieve acceptable solutions cannot be specified in advance and where it is difficult to say which user steps are directly relevant to the quality of the solution[1]. Examples of such tasks are design problems where the user may get support in organizing the design and evaluate different design steps. Planning and allocation problems are other examples[2]. Adequate information systems for this kind of tasks should emphasize support of the activities of the early stages of problem solving, where the problem is conceptualized, the underlying ideas are created, reformulated, modified and discarded.

The limitations of analytical methods of mathematics are not the only or the most essential motivation for the growing awareness in recent years of the possibilities of exploiting the computer in tasks of this kind. Rather, this awareness expresses a new way of looking at the relationship between the user and the computer, where man's creative ability and knowledge are taken advantage of in problem solving situations and where his task is not reduced to input data in a prescribed way. To our judgement this way of using computers

---

[1] Bennett, John L. (1983), Analysis and Design of the User Interface for Decision Support Systems. In Bennett, John L. (editor), *Building Decision Support Systems*, Addison-Wesley Publishing Company, Reading.

[2] Garey, Michael R. and Johnson, David S. (1979), *Computers and Intractability*, W. H. Freeman & Co, San Fransisco.

will become increasingly important and will be one alternative, which can counteract the impoverishment of the job content which in many cases is connected with traditional computerized administrative systems.

Crucial for this new kind of computer applications is that they should admit the user to intervene into the course of the computations. They should support a tentative way of work and allow for testing and developing possible partial steps on the basis of achieved results. This includes activities to accomplish a more precise problem description by identifying relevant components and their relationships, to develop new concepts and ideas and define their relationships, to develop models describing certain aspects of the problem domain, to derive constraints on the variables of the model, and so on. In addition to structuring activities of this kind the user often makes evaluations of partial steps in order to decide how to proceed, e.g. testing the outcomes of a model for different values of the parameters. The progression of work may develop along several alternative lines, which the user has to keep track of and compare. During this process it is often necessary to be able to swiftly change the focus of attention and concentrate on the currently most important aspects. We call computer systems aiming at realizing such an environment *interactive information systems*[1].

The interactiveness stresses problems concerning the role of computer applications in purposeful human activity and how the use of computers is integrated into the user's practice. The use of the term *information system* stresses that the intentions of the user and the kind of application are important in the design and use of information systems. This is in contrast of, for example, those who state that the human-computer interface can be studied independent of the application. The term *information* also indicates that we are interested in applications which involve problem solving or design tasks. This will lead us to show proper concern to cognitional and epistemological questions in the use of computer applications.

2. On a Theoretical Basis for Design

In this context a fundamental question concerns the theoretical basis on which to found the derivation of suitability criteria for an interactive information system. In contrast to more standardized computer applications the course of actions performed by the user

[1]Nilsson, Kenneth (1987), *Project Description: Design of Interactive Information Systems*, RRIPCS-5.89, Institute of Information Processing, University of Umeå, Umeå.

cannot be determined in advance. This means that in order to be helpful the interactive information system must in general be flexible. It is important that the user is not hampered by the absence of adequate means of expressions on behalf of the information system. On the other hand a flexible system does in general put requirements on the user's competence. There is a risk that the user's attention is drawn from the proper contents of his task in favor of dealing with the technicalities of the computer. An important research issue is therefore to find criteria for balancing between these two extremes.

From an engineering point of view it is often recognized that it is necessary in software design to take into account the limitations of human performance in addition to the functional properties of the software. The introduction of psychological aspects in this context is in many cases limited to the capabilities of humans as biological beings. Our position is that the introduction of psychological aspects in this respect is not enough. Some problems in designing information systems related in an earlier report give indications in this direction.[1]

A common trend in human-computer interaction research is the metaphor approach. The use of metaphors means that in designing a computer application one tries to choose a domain familiar to the user and associate different objects and actions of the computer application with corresponding objects and actions of the familiar domain. A well-known example is the desktop metaphor (Here files correspond to documents, file directories to document folders, a paper trash is used when you delete files and so on). One often takes a pragmatic approach and tries to connect to the user's everyday experience. The ties between the computer application and the corresponding familiar domain can be rather loose. One rationale for this can be that the application is not very well known or hard to predict when the computer system is designed. A dilemma in applying metaphors is the abundance of analogies that may be found between the computer objects and the corresponding objects of the supposed familiar domain. Also in this case a more profound theoretically based analysis is called for.

We claim that a more profound analysis of the way in which computers are used as a part of human work is necessary. It is crucial to find a theoretical basis for better understanding human activities, in which computers can be used, and the involvement of computers in these activities. The need of such a new understanding reflects a transition in the way of conceiving the role of the computer from a machine for accomplishing separate clearly defined computational tasks to a means or tool to be integrated in a human

[1] Nilsson, Kenneth (1984), *Några problemställningar kring datamodellering och interaktiva databastillämpningar* (Some problems on data modeling and interactive database applications, in Swedish), report UMADP-WPIPCS-1.84, Institute of Information Processing, University of Umeå, Umeå.

activity or work process. This transition brings to the fore questions, such as what we mean by knowledge, the possibilities and use of representations of knowledge, the role of the social context in which computers are used, etc.

An important step in this direction has been taken by Winograd and Flores[1] in their criticism of the underlying assumptions of what they call the 'rationalistic tradition' prevalent in cognitive science and computer science. They object to the view on knowledge as representations that can be used to model intelligent behavior and thus that a computer program can be intelligent and understand natural language. The rationalistic tradition has a tendency of equating human thinking with the manipulation of mental representations of knowledge, i.e. mental operations are assumed to be analogous to computer models of reasoning. This misunderstanding of human thinking constitutes a hindrance in designing effective computer systems for supporting the interaction of human beings and computers.

3. Winograd and Flores

Winograd and Flores propose an alternative theoretical basis for design of computer based systems. Instead of an individual-centered conception of knowledge and understanding resulting from formal operations on mental representations they advocate as a basis for design the individual's committed participation in mutually oriented patterns of behavior that are embedded in a socially shared background of concerns, actions and beliefs[2]. Inspired by Martin Heidegger they say that conscious reflection and systematic thought are secondary to pre-reflective experience of being thrown in a situation in which we are already acting. We are always engaged in acting within a situation, without the opportunity to fully disengage ourselves and function as detached observers[3]. This 'being-in-the-world' (Heidegger) or 'structural coupling' (a term from the biologist Humberto Maturana, another source of inspiration for Winograd and Flores) is thus essentially an unarticulated relationship between us and the world. Representations arise in *breakdown* situations, i.e. the interrupted moment of our habitual, standard, comfortable 'being-in-the-world'[4]. Things and their properties emerge in breakdown,

[1] Winograd, Terry and Flores, Fernando (1986), *Understanding Computers and Cognition — A New Foundation for Design*, Ablex Corporation, Norwood NJ.

[2] *ibid.* p. 78.

[3] *ibid.* p. 71.

[4] *ibid* . p. 77.

when we unconceal them as a matter of concern. They are conceptualized to help anticipate and cope with breakdowns. Language needs express only what is not obvious and the need of articulation is dependent on a shared background between the speaker and the listener. If there is a small degree of shared background the language has to be more articulate than between people who to a great extent have a common experience. What is required is sufficient coupling so that breakdowns are infrequent and a standing commitment by both the speaker and the listener to enter a dialog in the face of breakdown[1].

Winograd and Flores denote as naive the view that we codify and store our experiences as representations in the brain. They refer to Maturana, who states that living organisms can act adequately in their medium without having a representation of their environment in their nervous system. Through 'structural coupling' an organism comes to have a structure that allows it to function successfully within its medium. This correspondence is indirect as created by the result of actions produced by the organism and the demand to maintain its function for survival. In a similar manner the correspondence between language and our non-linguistic medium is equally indirect[2]. Knowledge is *always* the result of an interpretation, which depends on the entire previous experience of the interpreter and situatedness in a tradition[3]. A representation can thus be considered as an interpretation of the history of a certain context in which breakdowns have occurred, the history which gives meaning to what we say in this context.

What do the above observations mean to designing computer applications? We will in the following paragraphs refer to some of the conclusions by Winograd and Flores. One conclusion concerns knowledge representations in computer programs. When representing knowledge in a formal system only those aspects of the background that can be taken into account are those that are explicitly represented. If a breakdown occurs which is not anticipated by the designer, the program has no possibility to go beyond the initial formalization to respond adequately to the situation[4]. This would require a reinterpretation with respect to a context that the computer program has no access to. As humans, when we are reasoning, we will do reinterpretations against an unformalized background. This is a much more flexible and powerful basis for acting than resorting to a pre-defined representation. In fact, this ability can be considered as a characteristic of intelligent behavior. This means that computer programs are in principle unable to exhibit intelligence or understand natural language. Another difference between computers and

[1] *ibid.* p.63.

[2] *ibid.* p. 62.

[3] *ibid.* p. 74.

[4] *ibid.* p. 75.

humans pointed out by Winograd and Flores is the ability of humans to enter into commitments and to be responsible for the courses of action they anticipate. This ability is also an essential part of intelligent behavior[1].

Our second remark relates to the role of breakdowns as an objective for design. A breakdown does not necessarily imply something negative to be avoided. It can mean that we discover a new important aspect of the activities we are engaged in to accomplish and the using of tools in these activities. An essential part of designing a computer system is then to anticipate recurrent breakdowns and provide the computer program with an adequate repertoire of aids to be used when they occur. These breakdowns do not have to be related to the computer only but can also pertain to the application domain. The objects and properties that constitute the domain of action for a person emerge in breakdown and are thus at the heart of design[2].

A third point is the possibility to define systematic domains. The fact that we commonly use a word does not mean that there is an unambiguously formal definition of its meaning. But recurrent patterns of breakdowns make it possible to define systematic domains, in which definitions and rules are articulated. Systematic domains can be the basis for developing computer programs that serves as effective mechanisms for the manipulation of formal representations. As examples Winograd and Flores give accounting programs, word processors and expert systems. The systematic domain of a word processor is the typographical aspects of language, incorporating such things as letters, words, punctuation, sentences, page layout, type fonts, and so on. A word processor does not 'understand' language but its systematic domain has an adequate correspondence to the activities which users of word processors find meaningful[3]. Knowledge acquisition in designing expert system can be considered as defining a systematic domain, discovering what is regular in some part of a professional's work.

Finally, we take up a phenomenon, that Winograd and Flores call blindness. When we analyze a situation in terms of objects and their properties we create blindness. We are limited by what can be expressed in the terms we have adopted[4] . In their explanation they refer to Heidegger who states that objects and their properties are not inherent in the world. When we use a hammer as a tool we do not think of it as an object. It is a part of the background taken for granted and is in a state of being ready-to-hand to us. It is only when a breakdown occurs, e.g. when the handle of the hammer breaks, that we are made

[1] *ibid.* p. 106.

[2] *ibid.* p. 165 - 166.

[3] *ibid.* p. 175.

[4] *ibid.* p. 97.

conscious of the hammer as an object. It then becomes present-to-hand to us[1]. We can conceive our experience as involving present-to-hand objects and properties from our pre-conscious experience of them as ready-to-hand.

4. Creativity

The views on cognition and understanding in connection with the use of computers put forward by Winograd and Flores, which is briefly accounted in the preceding section, seem to contain a number of relevant aspects for designing interactive information systems.

Earlier in Section 1 we contrasted interactive information systems with computer applications aiming at automating human intelligence. This automation presupposes that it is in principle possible to represent a complete description of human cognition as a formal system. Winograd and Flores give good reasons for that this way in the spirit of the 'rationalist tradition' is wrong due to the limitations of formal reasoning. Their view gives a motivation for an interactive approach based on an alternative understanding of human cognition and action. Of special interest for interactive applications is the possibility to isolate aspects suitable for defining systematic domains and the possibility for interaction between the systematic and the contextual. It is then possible to define effective mechanisms for manipulating formal representations, while the meaning of the terms is interpreted by the user with respect to the context he or she is working in.

Interactive information systems are supposed to be helpful in the early stages of a problem solving process. It is then important that computers can be helpful in creative work. In the following we will give some remarks on how to characterize creativity. We will then in the following section introduce some aspects of the epistemology of the French philosopher Gaston Bachelard. His ideas represent in this context an interesting contrast to Winograd and Flores concerning the role of theoretical reflection and formalization versus practical understanding and common sense knowledge. In the last section we will discuss some consequences of this difference.

Creativity can be defined as the ability of an individual to come up with new and original products and ideas. We call a person creative if he or she has unusually many such ideas. When we say that an idea is original, we can mean that it is a new idea that nobody has thought of before or that a person has not got his idea from anywhere else. However,

1 *ibid.* p. 37.

there is also a more narrow sense of the term. Among the ideas in the broader sense, some strike us as more inventive, surprising or imaginative than others. Other ideas although they are not borrowed seem obvious, prosaic or commonplace. The latter category is not considered original in this narrower sense of the word[1]. To sum up, we can say that an original idea is one that creates a break of continuity with respect to the prevalent ideas in a certain domain. The most notable examples should be in art and scientific work.

5. Bachelard's epistemology

In this section we will give an account of some aspects of the epistemology of the French philosopher Gaston Bachelard. Usually, philosophy of science and epistemology preoccupies itself with the task of establishing conditions for the validity of knowledge. However, in this context it is not the question of validity that is of most concern to us. Rather we are interested in a better understanding of the process in creating new knowledge and how to surmount the obstacles we can encounter and thus further creativity. Bachelard works out a dynamic, dialectical concept of science, where he stresses science as a process, comprising corrections of wrong assumptions, breaks with earlier theories and eliminations of contradictions. The crucial aspects of the research process will then be the transition periods when the meaning of the fundamental concepts changes.

Gaston Bachelard (1884 – 1962) has presented his epistemology in a number of works[2]. However, the presentation here is based on the introduction to Bachelard by Dominique

[1] Goldman, Alvin L. (1986), *Epistemology and Cognition*, Harvard University Press, Cambridge MA., p. 248.

[2] Some of the most important works in which Bachelard presents his epistemology are

Bachelard, G (1934), *Le nouvel esprit scientifique*, PUF., Paris,

Bachelard, G (1938), *La formation de l'esprit scientifique*, J. Vrin, Paris,

Bachelard, G (1938), *La philosophie du non*, PUF., Paris,

Bachelard, G (1949), *Le rationalisme appliqué*, PUF., Paris,

Bachelard, G (1953), *Le matérialisme rationnel*, PUF., Paris.

Lecourt[1], whose writing on Bachelard's work is one of the most penetrating and comprehensive.

Bachelard engaged himself in the study of the rapid and revolutionary development of physics in the first decades of this century[2]. What drew his attention was the novelty of these theories and the concepts they brought into play. Bachelard opposes the empiristic, 'common sense' conception that theories are inductive hypothesis of given data based on a non-theoretical experience. Scientific theoretic knowledge does not arise as spontaneous synthesis of given observed data. On the contrary, scientific knowledge is a produced reality. Bachelard emphasizes that the scientific knowledge is essentially different from everyday knowledge. This means that scientific knowledge cannot be explained as a continuation of pre-scientific everyday knowledge. The concept of epistemological rupture (*rupture épistemologique*) is used to express this disengagement from the immediate experience in favor of scientific constructions as well as to express the fact that science progresses discontinuously. As a consequence of this discontinuity science, in a sense, has no history. "There is no transition between Newton's system and Einstein's system. One cannot get from the former to the latter by collecting knowledges, taking double pains with measurements, slightly modifying principles. On the contrary an effort of total novelty was required."[3] Rather, it is the present that explains the past, Einstein that explains Newton.

Everyday experience and pre-scientific knowledge can however act as what Bachelard calls epistemological obstacles, which if not eliminated will hamper the development of science. More generally, epistemological obstacles will appear any time whenever an existing organization of thought — scientific or not — is in danger. The obstacle is solidary with a determinate structure of thought which will later appear, by recurrence, as a 'tissue of tenacious errors'. An obstacle can arise at the moment of the constitution of the knowledge, in the term of a 'counter-thought' or at a later phase in its development as a suspension of thought.

The natural tendencies of the mind, according to Bachelard, are anti-scientific and science is a constant struggle against these tendencies. As an agent of the process of scientific practice, the scientist has to submit to the norms of this process. If he refuses he puts himself outside science. But he also has to make an effort to get there, to make a break with the spontaneous interests of life. He has to break with the reverie of everyday

---

[1] Lecourt, D (1975), *Marxism and Epistemology — Bachelard, Canguilhem, Foucault*, NLB, London 1975. (this book consists essentially of a translation from French of *L'épistemologie historique de Gaston Bachelard*, Vrin, 1969 and *Pour une critique de l'épistemologie*, François Maspero, 1972).

[2] The genesis and conspicuous successes of the quantum theory and the theory of relativity.

[3] Bachelard, G. (1934), *Le nouvel esprit scientifique*, PUF, Paris, 15^{e} ed. 1983, p. 46.

experience and mode of thinking. This is the object of art and poetry, while science is the domain of reason. The scientific character of science is guarantied not by its foundations in reality, but by the social character of its institutions. The distinction between science and art does not mean that rational or logical thinking should have a monopoly position. Science and art are developed by contrasting them, like reason and phantasy. They have a common origin in the creative activity of the mind. Science itself presupposes the ability of imagination[1].

Bachelard ascribes to mathematics a more prominent role than being just a language of formalization. Its essence lies in its power of invention. Mathematics is the driving element of the dynamism of scientific thought and functions as an instrument for working out theories. Bachelard asserts that the *extension* of a concept has precedence over its *comprehension.* Rather than seeing in a word a *being* the scientist sees in it a *concept* all of whose being is resolved in the system of *relations* in which it is inscribed. It is extension that illuminates comprehension[2]. 'It is by extending an idea to extremes that one grasps its maximum comprehension'[3].

Bachelard works out a dynamic dialectical concepts of science. Science proceeds as a historical process of mutual adjustments between theory and experiment. The sciences must incorporate a phenomeno-technics, a technique for the production of phenomena, as well as concepts. Indeed a concept becomes scientific only in so far as it is accomplished by a technique of its realization. Conversely, and in opposition to the thesis that theories are instruments for describing and predicting experience, Bachelard insists that instruments are materialized theories.

6. Discussion and Conclusions

It may seem inappropriate to introduce Bachelard in this context, since his ideas might be too far away from the ideas of Winograd and Flores, which we have discussed earlier. However, there are some affinities between them that involve ideas fundamental for both of them. Bachelard's rejection of the view of scientific discovery as a process whereby

[1] We have here described only one side of Bachelard's work. The other side represents comprehensive writings on poetry.

[2] A similar position is taken by C. West Churchman. In an example of planning he illustrates this by means of a type of planner whom he calls the 'ideal planner'. The ideal planner sees planning as a never ending process, as an unfolding process, where each new step could contribute to a deeper insight on behalf of the planner. Churchman, C. West (1979), *The Systems Approach and Its Enemies*, Basic Books, New York, p. 83ff.

[3] Bachelard, G (1949), *Le rationalisme appliqué*, PUF., Paris, p. 94.

new knowledge is assimilated into a system that changes only in so far as it grows is in the spirit of Winograd's and Flores' critique of the rationalistic tradition. Also, Bachelard's attribution to the error the role as an essential, necessary and driving moment of knowledge instead of an accident on the road seems close to the concept of breakdown that Winograd and Flores use[1].

However, the prime rationale for introducing Bachelard's epistemology here is that it illuminates certain aspects of creativity that do not seem to fit in the framework of Winograd and Flores. Despite the distinction that Bachelard makes between ordinary and scientific modes of thought it seems reasonable that the conditions of scientific work can be extended to creative work. In both cases there is a demand of creating something novel, that requires an extra effort from the persons involved. Intuitively, also the occurrence of obstacles appears to be transferable to the case of creative work. Of special interest is the possibility that immediate experience and existing structures of thought can be a hindrance to creative work. This is a kind of blindness that is reversed to the blindness Winograd and Flores speak of.

We will now return to Winograd and Flores and take up some aspects on their view of human cognition. Inspired by Heidegger they ascribe to reflection a hidden away role in the process of cognition[2]. The following statements indicate this. Knowledge lies in the being that situates us in the world, not in a reflective representation[3]. One of the most fundamental aspects of Heidegger's discourse is his emphasis on the state of thrownness as a condition of being-in-the-world. We do at times engage in conscious reflection and systematic thought, but these are secondary to the pre-reflective experience of being thrown into a situation in which we are already acting[4]. We have primary access to the world through practical involvement with the *ready-to-hand* — the world in which we are already acting unreflectively[5]. As one conceives breakdown as presented in the book of Winograd and Flores, one gets the impression that breakdown mainly occurs as a result

---

[1] Another perspective on the role of error in developing new knowledge is given by Kristo Ivanov in his thesis Ivanov, Kristo (1972), *Quality-Control of Information — On the Concept of Accuracy of Information in Data-Banks and in Management Information Systems*, Royal Institute of Technology, Department of Information Processing and Computer Science. Stockholm, p. 4.36, 4.44 ff and in Ivanov, Kristo (1986), *Systemutveckling och rättssäkerhet*, (in Swedish), Svenska Arbetsgivareföreningen, Stockholm, p. 46 - 52.

[2] This observation has also been put forward by William J. Clancey, in a review of Winograd and Flores book, *Artificial Intelligence*, vol. 31 (1987), p. 246ff.

[3] Winograd, Terry and Flores, Fernando (1986), *Understanding Computers and Cognition — A New Foundation for Design*, Ablex Corporation, Norwood NJ., p. 74

[4] *ibid.*, p.71.

[5] *ibid.*, p. 32.

of something happening around us. Our role as interpreters of the medium we live in is emphasized, while our ability to take initiatives to change it is not very salient.

We learn from Bachelard the inventiveness of mathematics as an instrument for working out theories. It is not too daring a hypothesis that the computer with its ability to manipulate formal systems could serve as an inventive instrument and help us in many situations where where we need to reflect or be creative. For example, when we want support in representing relationships between concepts to organize our thoughts or to generalize our ideas, or when we want to simulate what might happen in a model, to visualize certain outcomes and so on. The computer might help us break our ingrained thoughts and perceptions to see new things.

The approach of Winograd and Flores with its foundation in practical understanding gives a strong support to the tool metaphor, i.e. the computer is conceived as a tool like a hammer. The ideal situation is then when we are not aware of using the computer, when it is ready-to-hand to us. In many applications this is certainly a most adequate metaphor. However, there is a risk that we confine ourselves to uses of the computer that fit into this metaphor, neglecting that there are also many applications where we need to be more conscious about the dialectic of practical understanding and reflection[1].

What is inspiring about Winograd's and Flores' approach is its attempt to overcome the standpoint of the rationalistic tradition, that in computer system design all aspects of reality can be caught in formal representations and their concern to gain a better understanding of the role the computer can play in human activities. However, in the analysis above we found that the use of the computer as an instrument in creative work did not seem to fit into the theoretical framework of Winograd and Flores. We need a theoretical basis that gives full recognizance to the interplay or dialectic between reflection and practical understanding in order to design information systems that support creativity. We have used Bachelard to point out certain aspects of creativity. Although there are certainly further aspects that could be exploited, the epistemology of Bachelard might yet not be the adequate theoretical basis for computer systems design. His object of research is primarily the physics of the first decades of this century and it is hard to say to what extent his ideas can be generalized. Also, he does not treat the question of how epistemological obstacles arise and how they are formed.

[1] Some researchers have pointed out that an important task of a system designer is to bring in new ideas among future users, that they can exploit when participating in systems development projects, e.g. Ehn, Pelle (1988), *Work-Oriented Design of Computer Artefacts*, Arbetslivscentrum, Stockholm and Bødker, Susanne , et al. (1988), Computer Support for Cooperative Design in *Proceedings of the Conference on Computer-Supported Cooperative Work,* Portland, Oregon.

Winograd's and Flores' emphasis on pre-understanding and unreflected action and neglect of the importance of reflection in human cognition also come into evidence in that they in their discussion of cognition make no qualitative distinction between humans and other living organisms. Perhaps a more fruitful approach would be to resort to the theory of activity founded by Vygotsky and Leontiev[1]. This theory has in common with Winograd and Flores the emphasis of the roles of actions in cognitive development but postulates a decisive difference between humans and animals concerning the mechanisms behind cognition[2]. In the activity theory work plays a specific role in developing man's higher mental abilities and in mediating the experiences from generation to generation. Characteristic of work is the use and production of tools on one hand and the completion of the work process under the shared conditions of a collective activity on the other. The tools mediate activities which relate a human being not only to the nature and objects around him but also to other members of his society. In this way he will receive experiences from earlier generations. This mediation can take place only in an external form such as material objects, actions or language. The higher human mental processes can therefore develop only as a result of an interaction from man to man. The historically created human properties, abilities and patterns of behavior will in an act of acquisition be reproduced by the individual. By the acquisition the individual completes what the animal achieves through heredity. The evolutionary, selection dependent, genetic development will thus be abolished and transformed through the accumulation of societal experience.

Contrasting the approach of Winograd and Flores with the epistemology of Bachelard and the human activity approach has thus revealed some observations with respect to creativity that are important in our further research for a theoretical basis and operational methods for the design of interactive information systems.

---

[1] Susanne Bødker has introduced this theory in connection with computer system design in her thesis Bødker, Susanne (1987), *Through the Interface — a Human Activity Approach for User Interface Design.* DAIMI PB - 224, Aarhus University, Aarhus. Another important source of inspiration for Bødker's thesis is the book of Winograd and Flores (*ibid.* p. 45). Concerning the differences between Leontiev and his followers on one hand and Winograd and Flores on the other with respect to the role of unreflected action and reflection in human cognition, to my judgment Bødker interprets Leontiev in a way which is in accordance with Winograd and Flores. She is critical to some authors within the human activity approach that take a stance in discussing automation of human operations by machines that according to her is similar to that of cognitive science (*ibid.* p. 52). She writes: "This view is closely related to the tendency in Marxist thinking of considering so called scientific or theoretical knowledge, i.e. breakdown knowledge, as superior or more profound than everyday knowledge and practice. " (*ibid.* p. 52).

[2] Leontjew, A.N. (1977), *Problemer i det psykiskes Udvikling*, Rhodos, Copenhagen, (in Danish), p. 623 ff.

Organisational Culture, Cooperative Work and Information Technology

A personal quest for an organisational framework

© *Agneta Olerup*

Project SOFIA
Dept of Information & Computer Science, Lund University
Sölvegatan 14a, S-223 62 LUND, Sweden
e-mail: adbao@seldc52

Abstract
The paper briefly presents CSCW and cooperative work. Assumptions in theories of organisations are listed. Organisational culture is discussed in terms of corporate culture, culture as value-systems and culture as meanings. Frameworks integrating structure and meaning are presented. Finally, a provisional organisational framework is put forward.

Introduction

Information technology and computers have for a long time been used to improve organisational efficiency. Information technology, desk computers etc, have been introduced for individual use with the aim to increase individual productivy. Recently various computer tools and facilities for cooperative work have appeared. At the same time there are increasing demands for job satisfaction, challenge and learning on jobs; as well as for democratic or group-oriented work-organisations.

Reviews of empirical studies on the impact of computers (e.g. Attewell & Rule 1984) have not found any consistent findings, nor have the studies been based on similar models of hypothesized relationships. The basic methodological assumptions in most empirical studies imply *technological determinism*, some have assumed *organizational implications* on technology, and only a few have regarded *information technology and organizational factors as interacting* (Markus & Robey 1988).

Clearly, broader and more comprehensive frameworks are required, considering a broader area of the organisation and also using more complex models. Information technology needs to be seen as one of several factors co-producing the outcome.

The evidence on what happens in organisations when some computer facilities have been installed is not systematic. Anecdotal evidence, however, suggests that in some companies the installation is accepted, and used but no more, while in other companies the installation has stimulated further development controlled by the users. Obviously this cannot be explained by looking only at information technology, organisational and individual factors, contextual factors or the organisational control system (broadly). It does suggest that leadership and organisational culture, and various sub-cultures, are important factors. Another important and closely related factor is the meanings of information technology and applications to the individuals in the organisation.

In this paper the focus is on computer supported cooperative work and organisational culture. The paper attempts to explore, in a very preliminary and provisional way, some features of an organisational framework integrating organisational culture and CSCW. The framework is intended to be used for studying the role of organisational culture with regard to cooperative work and CSCW.

Computer Supported Cooperative Work – an elusive field?

Computer supported cooperative work, for which the acronym CSCW has been widely adopted as a label, is a fairly new term, but it is receiving increasing attention from researchers, industry and international professional organisations; conferences are being organized regularly by various bodies, the number of publications (conference-papers and journal-artciles) are increasing rapidly. Reviewing the published material one receives an impression of confusion and ambivalence.

In an excellent paper Bannon and Schmidt (1989) clarify several issues with regard to Computer Supported Cooperative Work and the acronym CSCW. Bannon and Schmidt (1989) convincingly argue that CSCW is a research are aimed at the design of application systems, united by the support requirements of cooperative work. Thus it is essential to understand the nature and characteristics of cooperative work with the goal of designing adequate computer support. The term "cooperative work" has a sound pedigree, dating from the first half of the 19th century (Bannon & Schmidt 1989). There are many varieties and forms of cooperative work. Attempts to determine the meaning of cooperative work though simplifying, may be grouped into three groups: all work is cooperative, cooperative work needs to satisfy certain criteria, and the term cooperative work is inappropriate.

It is maintained that *all work is essentially cooperative* since it depends on others for its successful performance (Ehn according to Bannon & Schmidt 1989), similary others suggest that cooperative simply means socially organized and all work is considered to be so (Suchman 1989). Then the term cooperative appears redundant. The value of the term cooperative work, however, lies in raising the awareness of designers of the basic social character of work, involving a multitude of workers, also work procedures are not well-specified and they are not only developed as efficient and effective means to accomplish certain ends, instead they are socially constructed and reconstructed. On the other hand, Sørgaard (1988) proposes a set of very *specific criteria for what would count as cooperative work.* Only work that is non-hierarchical, non-specialist and relatively autonomous would in this view be considered cooperative. This comes very close to equating cooperative work and leaderless groups (e.g. Manz & Sims 1986, 1987). Cooperative work is regarded as inherently positive. Finally, it has been argued that the term "cooperative work" is *inappropriate* because of inherent idealogical overtones. Alternative terms put forward include collective work (Howard 1987), groups and groupwork (Johansen 1989).

Critically examining these, and other suggestions, Bannon and Schmidt (1989) argue that cooperative work comprises various modes of interaction: direct (involving interpersonal communication) versus indirect (mediated by the transformation process), as well as collective versus distributed. Concluding, the term "cooperative work" is the general and neutral designation of multiple persons working together to produce a product or service (Bannon & Schmidt 1989). Furthermore, cooperative work does not imply specific forms of interaction or organization, neither does it imply a particular degree of self-management nor a particularly democratic leadership style.

The term Computer Supported Cooperative Work can be subdivided into two subterms: Computer Support and Cooperative work. Work in the CSCW field seems to be oriented either towards computer support and having a technological orientation, or towards cooperative work and it is then oriented towards organizations, people and work. In the *technological* orientation the focus is on designing computerized facilities to support groupwork (Johansen 1989, Olson 1989). Groupware refers to software which can be used by groups for managing, planning and doing joint tasks, including support for face to face meetings, support for electronic meetings and support between meetings. In the *work* orientation the focus is on the characteristics of work and cooperative work and the requirements on computer support. The specific characteristics of cooperative work identified encompass (Bannon & Schmidt 1989): articulating (i.e. allocating and coordinating) cooperative work, sharing an information space and designing a socio-technical system (i.e. the joint design of computer support and organisation).

The CSCW field is largely oriented towards administrative and office work, thus an organisational framework needs to take into account the major characteristics of such work. The accepted view of office work assumes task certainty, common goals and formal procedures. It is however increasingly recognized that this provides a delusive and fictitious view of offices and office work. Instead, office work is characterized by task uncertainty and important roles are played by informal office practices. Office work is socially organized, office procedures are constituted by activities, which are negotiated and renegotiated with co-workers, thus they are created and recreated. Office procedures are products of the orderly work in the office. The meanings of activities and procedures are constructed in these processes, thus office work is socially constructed. Furthermore, there are no common goals. Instead offices are organisations characterized by a mixture of conflict and collaboration. They can be viewed as coalitions of individuals having parochial interests and aspirations and pursuing individual goals (Cyert & March 1963), or they can be viewed as political systems with scarce resources and conflicts among goals (Pfeffer 1981).

Organisations - mythical elephants?

Organisation theory, organisational behaviour or - recently - organisational science is a highly differentiated and heterogenous field. It encompasses a broad range of theories and models of organisations, drawing on various social sciences (e.g. anthropology, political science, psychology, sociology, economics). Theories of organisations help us in understanding various aspects of organisations, from various perspectives, object-levels and conventions. Taken together theories of organisations provide a kaleidoscopic view of organizations, but they don´t tell us what organisations are[1].

Table 1. A sample of theories of organisations

Classical School	Alternative theories
mechanical systems	organismic system (Katz & Kahn 1966, structural contingency theory [2])
negative feedback, correcting for deviations	positive feedback, learning (Katz & Kahn 1966, Maruyama 1963)
closed system	open system (Katz & Kahn 1966, Thompson 1967)
organisational structure is static, hierarchical, based on decomposition into constitutent parts	structure is dynamic, cycles of activities and events, interactions, ongoing and changing (Katz & Kahn 1966, Buckley 1967)
formal organisation	informal organisation (Human Relations)
universal principles for organisational structure	organisational structure is contingent on technology, environment and other contextual factors (e.g. structural contingency theory)
organisations have goals	organisations struggle for survival in changing environments (e.g. structural contingency theory)
organisational changes are goal-oriented	organisations need to manage organisation and environment relationships, environmental/contextual changes create pressures for organisational changes (e. g. structural contingency theory)
organisational change is goal-oriented and planned	change is internal, occassional and temporary; minor adjustments of daily routines; changes are small, unplanned and often neither purposive nor controlled, changes are enacted by the informal organisation (e.g.Christensen & Molin 1983)

organisations have common goals	organisations are collections of individuals engaged in purposive goal-taking behaviour; there is no unity on goals and objectives, organisations are political systems (Cyert & March 1963, Pfeffer 1978, Pfeffer 1981, Pettigrew 1973)
organisations consist of multiple parts, finely tuned towards the accomplishment of tasks	organisations are loosely-coupled systems, they remain stable despite numerous pressures for changes, changes are accommodated without requiring any major restructuring (March & Olsen 1976, Weick 1979)
organisations are rational	organisations are irrational (Brunsson 1985), ambiguous (March & Olsen 1976)
organisations are rational, and there is an objective reality	individuals and groups interpret goals, experience actions and outcomes subjectively. Action-oriented, interpretivistic approaches and social conscrutivism are crucial in understanding organisations (Berger & Luckmann 1967, Buckley 1967, Silverman 1970, Weick 1979)

Taking classical organisation or management theory as a starting point I shall briefly indicate some alternative organisation theories (see table 1). Table 1 by no means provides an exhaustive inventory of organisational theories. I shall not present the various theories, since that would require at least a textbook.

The classical school has been criticized in many respects, but it is still very strong with many advocates (e.g. Langefors 1970). Also, whenever organisations are described in terms of organisational schemas, allocation and coordination of tasks, the classical school is being used.

There is a great variety among theories of organisations in terms of what object-level they are oriented towards[3]. Some, based on interpretivistic ideas and social constructivism, are concerned with actions, individuals and groups, others are concerned with the organisation as a whole (e.g. structural contingency theory).

The mainstream of structural contingency theory (distinct from contingency theory of leadership) has inquired into the existence of organisation and environment relationships. Little attention has been paid to how correspondence between organisation and environment has been achieved. However, the notions of strategic choice and dominant coalition have been introduced in attempting to explain organisational adaptations of relationships with the environment (Child 1972, Hage 1977). Early contingency studies usually considered only one contextual/environmental factor, later studies have realized that contextual factors are mutually constraining and interdependent, thus they tend to form closely knitted nets, or archetypes. This facilitates formulating design theories, integrating the findings from structural contingency theory (e.g. Galbraith 1973, Kotter 1978).

Organisational culture

Organisational culture is a new and expanding field within organisation science. Reviews of the area indicate several orientations, and an increasing variety (Alvesson 1989b, Alvesson & Berg 1988, Smircich 1983a). There is no single theory of organisational culture, but a broad range, from theories of corporate culture to symbolic theories. Alvesson and Berg provide an insightful and useful survey of organisational culture.

There is no common definition of organisational culture; it is a jungle of definitions, concepts, theories and models, which are woolly, overlapping and unrelated. In order to put some resemblance of order into the field, Alvesson and Berg (1988) identify three dimensions: object-level, phenomena and conventions. *Object-level* refers to the extension or level of observation, *phenomena* to which aspects are in focus, and *conventions* are more or less explicit theoretical perspectives identifying the general character of object of research, which overall aspects need to be studied and fundamental scientific concepts, thus a convention provides general guidelines for definitions.

Within the field of organisational culture, five types of conventions may be distinguished: (1) cultural conventions, including among others corporate culture and culture as valuesystems, (2) culture as meanings, (3) ideology, (4) psychodynamic models, and (5) organisational symbolism (Alvesson & Berg

1988). In this paper I shall only briefly discuss corporate culture, culture as value-systems and culture as meanings.

In addition to the five types of conventions, a distinction is often made between culture as a variable and culture as a metaphor (Smircich 1983a). When culture is viewed as a *variable,* it is another component in organisations in addition to technology, goals, administrative systems etc; organisations have a culture. Culture as *metaphor* means a particular perspective on an organisation, every subsystem in an organisation is considered to be infused with culture. The metaphor perspective often involves studying traditional organisatinal, work design, economic etc issues but using an alternative theoretical frame of reference. In the metaphor perspective culture is a historical product, only changing slowly.

Corporate culture

Corporate culture is probably the best known sub-field within the field of organisaitonal culture. It has produced a large number of management-oriented books (e.g. Deal & Kennedy, Peters & Waterman), and there are close ties to classical management theory. Corporate culture is loosely defined in terms of common values, norms, assumptions and attitudes. Corporate culture is external and imported into an organisation, it can be manipulated. Corporate culture presupposes common interests, harmony and common goals (Alvesson & Berg 1988), thus it conforms to a managerial perspective.

There have been a number of cases, reported in popular business and management press, of radical changes in corporate culture, SAS is one case. An analysis of such cases indicates that the changes could much better be explained using traditional theories of strategic planning and design (Alvesson & Berg 1988). However, corporate culture provides a more attractive packaging for the strategic and organisational changes required.

As a managerial tool corporate culture is just another form of human relations, further implementing it into a company is often constrained by concerns for rationality and productivity (Mouritzen 1989).

El Sawy (1985) suggests an approach for cultural infusion when introducing information technology, since technical change often requires learning new practices that may threaten existing interpersonal relationships. Essentially the approach involves transferring values to users on information technologies in general and on particular applications (e.g. word processing is not necessarily a secretarial activity). There is hardly any argument that such changes as introduced by El-Sawy lack importance, but they primarily involve attitudinal changes.

Culture as value-systems

A number of researchers-consultants consider organisational culture as systems of values and articles of faith, emphasizing ideas common to those belonging to the culture. Probably the principal advocate for this convention is Schein (1985), who draws from social psychology, and defines organisational culture:

I will mean by "culture": a pattern of basic assumptions – invented or developed by a given group as it learns to cope with its problems of external adaptation and internal integration – that has worked well enough to be considered valid and therefore to be taught to new members as the correct way to perceive, think and feel in relation to those problems. (Schein 1985, p 9)

Thus culture is a historical product, and it does not refer to overt behaviour patterns, which are the outcomes of both cultural predispositions and situational contingencies.

Furthermore, three levels of culture may be distinguished (Schein 1985): (1) *artifacts and creations:* including technology, art, visible and audible behaviour patterns; this level is visible but often not decipherable; (2) *values:* which are testable in the physical environment, or testable only by social consensus; this level is characterized by a greater level of awareness; and (3) *basic assumptions:* relationships to environment, nature of reality, time and space, nature of human nature, nature of human activity, nature of human relationships (cf. Kluckhohn & Strodtbeck 1961); this level is taken for granted, it is invisible and preconscious.

The approach – involving participant observations, deep interviews, document analysis – aims to interpret and uncover deep-structures and basic assumptions, which often lie considerably underneath what is observed (Schein 1985).

Attempts have been made to operationalize Schein's model: (1) indicators for artifacts are physical arrangements, language, traditions and stories; (2) indicators for values are espoused values and values-in-use (Pedersen 1987).

Unfortunately, Schein (1985) is inconsistent and ambiguous with regard to the role of information technology in organisational change (pp 285-288), and inclined towards technological determinism (p 124). Also, the linkage between culture and organisational effectiveness hypothesized by Schein (1985) appears to be dubious, since other factors (e.g. technology, task uncertainty) are also operating (Alvesson 1989a).

Culture as meanings, social constructivism

This convention draws attention to the way experience becomes meaningful to a group of people. The focus is on how individuals make sense of – interpret and understand – their experience, and how these interpretations and meanings relate to action (Smircich 1983a, 1983b).

Very briefly, the basic ideas emphasize that people are active agents in creating and shaping reality. People enact their reality individually or in concert with others. Actions become meaningful in a wider social context, which can be understood as relationships between figure and ground, differentiating elements (a figure) from their wider context (the ground), and interpreted through a frame of reference providing coherence, social organisation depends on the emergence of shared interpretive schemes. Such schemes provide the basis for shared systems of meaning that allow day-to-day activities to become routinized and taken for granted. The stability of different modes of organisation depends on the on-going existence of common modes of interpretation and understanding. The impact of structures and behaviours is mediated through the meanings and interpretaiton they require (Berger & Luckmann 1967, Weick 1979).

Culture, in this convention, refers to how meanings are created, and the ways in which people interpret their experiences, and how the meanings relate to action. *Social structure,* on the other hand, is the pattern of actions and acts, the actually existing network of social relationships. Interpretations in terms of culture thus supplements social-structural analysis, and particularly focus on the meaning of organisational phenoma (Alvesson 1989b).

The analysis of organisations as cultures or sets of meanings must go beyond any single individual's understanding of situations. There is a need to be concerned with multiple meaning systems that may be in competition or conflict with one another (Smircich 1983b). Also, the analysis of meanings must go beyond the meanings of actions in terms of task accomplishment in a narrow (i.e. rational) sense.

Furthermore, recurrent themes that show how meanings and symbols are meaningfully linked and how they are related to the activities of the people in a setting, must be articulated (Smircich 1983b). The aim of analysing meanings is to see the world as people in the organisation see it, to learn the meaning of actions and events for the people in the organisation (Smircich 1983b).

The study of meanings has obvious theoretical connections with theories of social constructivism (Berger & Luckmann 1967), the action framework (Silverman 1970), theories on enactment (Weick 1979). Social constructivism is crucial in action-oriented systems theory (Buckley 1967) and in the theory of structuration (Giddens 1984).

Objectivity is a problematic issue in the study of meanings. According to social constructivism reality exists only as a common social construction, from which we see, interpret and act. There is no objective social reality in the same sense as a physical reality. An organisation can only be conceived as shared meanings of realities, where physical-objective conditions only provide the basis for these meanings but are of little concern apart from that. Reality becomes objectified, not objective (Berger & Luckmann 1967).

Realism (Sayer 1984, Silverman 1985) suggests that social structures are 'real' in the sense that they are partially independent of individuals and their perceptions.

Pro and con

Advocates of a cultural perspective suggest that the culture metaphor makes it possible to redefine traditional concepts of leadership and management (Bennis & Nanus 1985, Smircich & Morgan 1982). Furtermore, a cultural perspective may provide a richer understanding of organisational changes.

Critics argue that the culture perspective lacks substance. It is nothing but a more attractive packaging of well-known concepts and theories, such as Human Relations. This is certainly true of corporate culture.

On methodological issues, there is a crucial difference between the US and Europe. In the US criticisms of positivism and quantitative methods have gone hand-in-hand with the growth of the cultural perspective. The situation in Europe, and Sweden, is different (Alvesson & Berg 1988). Criticisms of positivism have a long standing in European, and Scandinavian[4], organisational and sociological research (e.g. Silverman 1970, 1985) and have prepared the way for qualitative methods.

Frameworks integrating structure and meaning

Frameworks resolving the conflict between structural approaches and approaches based on individuals and the meanings attached to situations are increasingly being proposed. In general, they conceptualize organisational structure as dynamic and on-going, organisational structure is produced and reproduced through human interaction (e.g. Barley 1986, Fombrun 1986, Ranson et al 1980)[5]. Similarly, within the philosophy of social sciences, realist approaches recognizes both the role of meaning in social processes and the constraining power of social structures (Silverman 1985).

Conceiving structure as produced by structuring processes attempts to integrate two notions of structure: structure as a formal configuration of roles and procedures, and structure as a pattern of social interactions (Barley 1986, Fombrun 1986, Ranson et al 1980). Structure is thus both a flow of ongoing action and a set of institutionalized roles and forms reflecting and constraining action. Structure is produced and reproduced through human interaction, simultaneously interaction is constained by structure. Furthermore, structure is unseperable from the experiences of individuals and their meanings.

Ranson et al (1980) suggest that three categories are integral in a model of the process of structuring: (1) *provinces of meaning* incorporating interpretive schemes and articulated values, (2) *dependencies of power and domination,* influencing relationships between interpretive schems and their relative importance, and (3) *contextual constraints,* in terms of organisational and environmental characteristics.

Barley (1986) is particularly interested in structuring, how it is occassioned and the forms it takes. Technology (in terms of equipment) provides an occassion for structuring; differentiated patterns of interaction may be distinguished as a group of people learns to cope with new equipment.

Fombrun (1986) emphasizes the dynamic aspects of structure, in addition to stability and homeostasis. Integrating individual actions involves processes of convergence and contradiction on three levels: (1) *infrastructure,* i.e.

constraints confronting an organisation, (2) *sociostructure,* e.g. administrative structures and managing exchange relationships, and (3) *superstructure,* i.e. the ideational side, meanings and interpretation.

The similarities and affinities between provinces of meaning (Ranson et al 1980) and culture as meanings are obvious. The importance of meanings in structuring processes are also stressed by Barley (1986) and Fombrun (1986).

The role of contextual constraints (e.g. environment, technology) in shaping structures are recognized, and contextual constraints are conceptualized in terms of pressures which are interpreted and acted upon by organisational members (Barley 1986, Ranson et al 1980). Similarly, Perrow (1970) stressed that technical uncertainty and complexity are social constructions which vary between settings.

Conceiving of contextual constraints in terms of their meanings for people is distinct from structural contingency theory, which considers contextual constraints as neutral and objective. Thus, care must be taken in drawing on the findings and organisational-environment relationships from structural contingency theory.

In summary, structure, culture and meanings as well as context are dynamically integrated in the frameworks put forward by Ranson et al, Barley and Fombrun.

Information Technology in a cultural perspective

Generally, introducing information-technology (including changes of information technology) into the work of people in an organisation disturbs ingrained patterns of interaction, and confirms other patterns of interaction. In such situations, individuals must learn new work practices, which may threaten existing patterns of interactions and meanings. Patterns of interactions are shaped and reshaped in intimate relation to a frame of reference encomapssing interpreted and reinterpreted meanings of both new and old technology as well as dito work practices. In this structuring process organisational structure is created and recreated. In addition to information technology other elements of the social context also influence patterns of actions and meanings of experiences and actions.

This very brief sketch above of the role of information technology in a social constructivist framework will suffice for this paper. A more extensive and detailed exposition would require using the terminology found in social constructivist and structuration literature - a terminology, which is unlovely, alienating and uncommunicative. Most importantly a theoretical presentation needs relevant empirical material to be comprehensible, otherwise it will probably just be a plagiarization with a change of terminology to conceal the plagiarization.

Still, this very brief sketch suggests that in a cultural analysis of information technology, a number of categories need to be considered: values, meanings, structure (as interactions) and context.

A proposed organisational framework

The organisational framework includes both the socio-structure and the culture in organisations. The aims are to obtain as rich a picture of what happens and goes on in an organisation, when information technology supporting cooperative work is introduced and used, both changes in work (and its organisation) as well as changes in meanings.

Important issues with regard to organisation and work are both the overall organisation and the work organisation. Cooperation and group-working (e.g. forms, extent, intensity) are some important characteristics of a democratic work organisation. The role of control and reward systems are important in establishing a democratic work organisation.

In addition, the context of an organisation or a work-unit provide important stimuli and pressure for the structure of tasks and the organisation of work. A number of contexts and forms can be identified: (1) the context of the *organisation as a whole*, (2) the context of a *particular work-unit* (e.g. work-group, a department), and (3) the *development* context, i.e. the context of the activities involved in developing a computer application.

Important in shaping the organisation of work and the use of information technology is the organisational culture, particularly values and meanings. In the study of values the framework put forward by Schein (1985) will be utilized, since this provides both a model of a value-system and a rigorous methodology for inquiring into value-systems. It has been argued that values and value-systems are important in connection with meanings (e.g. Ranson et al

1980). Social constructivism provides the theoretical basis for inquiring into meanings. It has been suggested previously in this paper that cooperative work is socially constructed, thus social constructivism provides means for linking cooperative work and organisational culture. The methodology of social constructiveness is, however, fairly elusive, but it does require intensive and extensive time-spending in the organisational unit in focus, for observations and conversations (i.e. living with the organisation).

The role of information technology in organisational changes is as one of several producers of organisational changes, and information technology is itself subject to change in the process. Thus information technology is a co-producer[6] in organisational changes.

The basic approach involves action research. The methodology combines qualitative and quantitative techniques, using quantitative techniques as support in a qualitative strategy. Further basic methodological principles are multiplicity and variety, i.e. collecting data from many sources using different data collection techniques and finally using different data analysis techniques.

Acknowledgements

In writing this paper I have benefitted from discussions with my co-researchers on project "SOFIA – Supporting Cooperative Work in Organisations with Future Information Technology", that is Sven Carlsson and Lars Fernebro (both at Dept of Information & Computer Science) and Ingvar Bokander (Dept of Psychology). I have had numerous discussions, mainly electronic ones, on culture and meanings with Dan Robey (Florida International University).

Project SOFIA has recieved funding (March 1989 - July 1991) within the programme "People, Work and Computers", which is jointly financed by the Swedish Work Environment Fund and the Swedish Board for Technical Development.

Notes

1. The heading to this section refers to the parable about the six blind men from Indostan and the elephant. The story-teller has the ability to see the whole elephant. This is in fact extremely arrogant (Churchman 1979). Analogously, organizational theories can be understood from the vantage point of an omniscient story-teller, or alternatively, there is no story-teller at all.

2. Elsewhere, I have reviewed structural contingency theory, as well as criticisms of the school (Olerup 1982), cf Child (1984)

3. Extensive, comprehensive and penetrating critical examinations of assumptions and methodologies in various organizational theories are provided by Burrell and Morgan (1979) and Perrow (1986)

4. Graduate courses in social science methodology offered by the Dept for Business Administration, at Lund University, during some years around 1970 included Berger and Luckmann (1967) as major course-assignment. (I was then a graduate/PhD-student in business administration at Lund University.)

5. Ranson et al (1980), Barley (1986) and Fombrun (1986) all make references to Giddens' theory of structuration (e.g. Giddens 1984). However, Willmott (1981) points out serious flaws in the way the theory of structuration has been used by Ranson et al (1980).

   (1) *Concept of structure.* Structure can be conceptualized as *interactions* (Ranson et al 1980, Barley 1986, Fombrun 1986), which is analogus to event-/activity-cycles (Katz & Kahn 1966). In the theory of structuration (Giddens 1984) structures are theorized as the *properties of social systems that are the medium and outcome of the practices that constitute social systems*, structures have three analytically separable dimensions (i.e. signification, domination, and legitimation) and interactions have three equivalent dimensions (i.e. communication, power, and morality/sanction) (Giddens 1984, cf Willmott 1981). Structure, in the theory of structuration and interactions can be compared to language and speech (Giddens, according to Willmott 1981). Structure, in Giddens', sense is not only more abstract than structure as interactions, it is also more abstracted, i.e. on a different level of analysis.

   (2) *Social constructivism.* The references to Giddens by Ranson et al (1980), Barley (1986) and Fombrun (1986) all concern the exposition of social constructivism by Giddens. However, other references are available, the major one is probably Berger and Luckmann (1967), and Buckley (1967) has integrated social constructivism into his social systems theory. Buckley (1967), quoting Selznick, traces the idea that structure is continuously opened up and reconstructed to among others G H Mead

   Comment. I find the frameworks proposed by Ranson et al (1980), Barley (1986) and Fombrun (1986) interesting, but they are not credible in their claims to be based on the theory of structuration. The fundamental notion in the theory of structuration is the concept of structure, so a framework claiming to use the theory of structuration, including social constructivism as used by Giddens, is only credible when structure is used in the sense of that theory (cf Willmott 1981). On the other hand, the theory of structuration is very intersting, but not easy to grasp, but it requires different approaches (Willmott 1981) than those proposed, for example by Ranson et al (1980).

6. Ackoff and Emery (1972) introduces a crucial distinction between cause-effect and producer-product. A *cause* is a *necessary and sufficient* condition for an *effect,* while a *producer* is *necessary but insufficient* for a *product.*

References

Ackoff R L, Emery F E (1972): On Purposeful Systems. Aldine-Atherton, Chicago

Alvesson M (1989a): Concedpts of Organizational Culture and Presumed Links to Efficiency. OMEGA Vol 17 No 4

Alvesson M (1989b): Ledning av Kunskapsföretag. Exemplet ENATOR. (Leadership in Knowledge-companies. The case of ENATOR.) Norstedts, Stockholm

Alvesson M, Berg P-O (1988): Företagskultur och Organisationssymbolism. Utveckling, teoretiska perspektiv och aktuell debatt. (Organisational Culture and Organisational Symbolism. Development, theoretical perspectives and current debate.) Studentlitteratur, Lund

Attewell P, Rule J (1984): Computing and Organizations: What we know and what we don't know. Communications of ACM vol 27 No 12

Bannon L, Bjørn-Andersen N, Due-Thomsen B (1988): Computer Support for Cooperative Work: An apprisal and critique. in Bullinger (1988)

Bannon L J, Schmidt K (1989): CSCW: Four Characters in Search of a Context. in Proceedings from EC-CSCW'89 (London Sep 1989)

Barley S R (1986): Technology as an Occasion for Structuring: Evidence from observation of CT Scanners and the social order of radiology departments. Administrative Science Quarterly vol 31 No 1

Bennis W, Nanus B (1985): Leaders. The strategies for taking charge. Harper & Row, New York

Berger P L, Luckmann T (1967): The Social Construction of Reality. A treatise in the sociology of knowledge. Penguin

Brunsson N (1985): The Irrational Organization. Irrationality as a basis for organizational action and change. Wiley, Chichester

Buckley W (1967): Sociology and Modern Systems Theory. Prentice-Hall

Bullinger H-J et al, eds (1988): Information Technology for Organisational Systems. EuroInfo 88. Elsevier (North-Holland)

Burrell G, Morgan G (1979): Sociological Paradigms and Organisational Analysis. Elements of the sociology of corporate life. Heinemann, London

Child J (1972): Organizational Structure, Environment and Performance. The role of strategic choice. Sociology (Clarendon) Vol 6

Child J (1984): Organization. A guide to problems and practice. Harper & Row, London

Christensen S, Molin J (1983): Organisationskulturer. Akademisk Forlag, Copenhagen

Churchman C W (1979): The Systems Approach. Rev ed. Delta, New York

Cyert R M, March J G (1963): A Behavioral Theory of The Firm. Prentice-Hall

Deal T E, Kennedy A A (1982): Corporate Cultures. The rites and rituals of corporate life. Addison-Wesley

El Sawy O A (1985): Implementation by Infusion: An approach for managing the introduction of information technologies. MIS-Quarterly Vol 9 No 2

Fombrun C J (1986): Structural Dynamics within and between Organizations. Administrative Science Quarterly Vol 31 No 3

Galbraith J (1973): Designing Complex Organizations. Addison-Wesley

Giddens A (1984): The Constitution of Society. Outline of the theory of structuration. Polity Press, Cambridge

Hage J (1977): Choosing Constraints and Constraining Choice. in Warner (1977)

Howard R (1987): Systems Design and Social Responsibility: The political implications of "Computer Supported Cooperative Work". A commentary. Office: Technology and People (Elsevier) Vol 3

Johansen R (1988): Groupware. Computer Support for business teams. The Free Press, new York

Katz D, Kahn R L (1966): The Social Psychology of Organizations. Wiley, New York

Kluckhohn F R, Strodtbeck FL (1961): Variations in Value Orientations. Row, Petersoln & Co, Evanston Ill

Kotter J P (1978): Organizational Dynamics. Diagnosis and interaction. Addison-Wesley

Langefors B (1970): System för Företagsstyrning. (Management Control Systems.) Studentliteratur, Lund

Manz C C, Sims H P jr (1986): Leading Self-managed Groups: A conceptual analysis of a paradox. Economic & Industrial Democracy (Sage) vol 7

Manz C C, Sims H P jr (1987): Leading Workers to Lead Themselves: The external leadership of self-managing work teams. Administrative Science Quarterly vol 32 No 1

March J G, Olsen J P (1976): Ambiguity and Choice in Organizations. Universitetsforlaget, Bergen

Markus M L, Robey D (1988): Information Technology and Organizational Change: Causal structure in theory and research. Management Science vol 34 No 5

Maruyama M (1963): The Second Cybernetics: deviation-amplifying mutual causal processes. Cybernetica Vol 6 No 1

Morgan G, ed (1983): Beyond Method. Strategies for social research. Sage

Mouritzen J (1989): Accounting, Culture and Accounting-Culture. Scand Journal of Management (Pergamon) Vol 5 No 1

Olerup A (1982): A Contextual Framework for Computerized Information Systems. Nyt Nordisk Forlag, Copenhagen

Olson M H, ed (1989): Technological Support for Work Group Collaboration. Lawrence Erlbaum

Pedersen J S (1987): Organizational Cultures within the Computing Field. Reflections on Schein's model of culture. CHIPS Working Paper 1987-2. Copenhagen School of Economics and Social Science

Perrow C (1970): Organizational Analysis: A sociological view. Tavistock

Perrow C (1986): Complex Organizations. A critical essay. 3rd ed. Random House, New York

Peters T J, Waterman R H (1982): In Search of Excedllence. Harper & Row, New York

Pettigrew A (1973): The Politics of Organizational Decision-Making. Tavistock

Pfeffer J (1978): Organizational Design. AHM Publ

Pfeffer J (1981): Power in Organizations. Pitman

Ranson S, Hinings B, Greenwood R (1980): The Structuring of Organizational Structures. Administrative Science Quarterly vol 25 No 1

Sayer A (1984): Method in Social Science. A realist approach. Hutchinson, London

Schein E H (1985): Organizational Culture and Leadership. A dynamic view. Jossey-Bass, San Fransisco

Silverman D (1970): The Theory of Organisations. A sociological framework. Heinemann, London

Silverman D (1985): Qualitative Methodology and Sociology. Describing the social world. Gower

Smircich L (1983a): Concepts of Culture and Organizational Analysis. Administrative Science Quarterly Vol 28 No 3

Smircich L (1983b): Studying Organizations as Cultures. in Morgan (1983)

Smircich L, Morgan G (1982): Leadership: The management of meaning. Journal of Applied Behavioral Science (JAI-Press) Vol 18 No 3

Suchman L (1989): Notes on Computer Support for Cooperative Work. Working paper WP-12, University of Jyväskylä, Dept of Computer Science

Sørgaard P (1988): A Discussion of Computer Supported Cooperative Work. Aarhus University. Computer Science Dept

Thompson J D (1967): Organizations in Action. Social Science bases of administrative theory. McGrawHill, New York

Warner M, ed (1977): Organizational Choice and Constraint: Approaches to the sociology of enterprise behaviour. Saxon House

Weick K E (1979): The Social Psychology of Organizing. 2nd ed. Addison-Wesley

Willmott H (1981): The Structuring of Organizational Structure: A note. Administrative Science Quarterly Vol 26

THE COMBINED EFFECT OF APPLICATION GENERATOR AND DEVELOPMENT STRATEGY ON DEVELOPMENT PRODUCTIVITY

Marikka Pihlaja
Helsinki School of Economics

Abstract: The promises of productivity improvements with application generators have been described by several authors. Also some experiments have been made, but they differ substantially in their research settings and conduct of the experiments. This paper examines these differences by gathering the main factors having an effect on the results of the experiments into a framework. Previous research efforts are then analyzed according to the framework.

By taking advantage of the developed framework experiments were conducted in order to evaluate the combined effect of application generator and development strategy on productivity. The results of these introspective cases showed that in comparison to third generation languages 30 to 1 productivity improvements in development process are possible, when application generators are used together with prototyping strategy. In addition, the programmers were more satisfied with their job in prototyping project than in linear projects.

1. INTRODUCTION

A severe problem facing data processing department is the backlog of applications. There are a number of studies which show the backlog to be several years long [Alloway & Quillard, 1983; Sumner & Benson, 1988]. Unfortunately maintenance of existing software requires most of the DP resources leaving only a small amount of resources for development of new applications. Therefore many useful new software products are not getting developed.

Methods of coping with the shortage of DP resources include the use of software packages and the support of end user application development. Despite the use of these methods the DP department is currently not able to satisfy the demand. Productivity of application development have become an issue that affects the profit and competitive ability of the organization.

Technological change and management of the new technology are the two basic means to achieve better productivity [Jeffery, 1987]. Application development is still highly labor intensive and since the costs of hardware resources are declining while the costs of labor keeps rising, the productivity will be improved by trading developer labor for more intelligent software [Boar, 1984; Boehm & Papaccio, 1988]. Repetitive and monotonous work phases can be automated or totally eliminated with the help of software tools [Boehm & Papaccio, 1988]. According to Boehm [Boehm, 1981; Boehm & Papaccio, 1988] productivity can be improved further by reusing components, building simpler products and eliminating rework with prototyping. In addition to these methods the management should pay special attention to human factors such as staffing and motivation [Boehm & Papaccio, 1988].

Significant productivity improvements have been associated with the adoption of fourth generation languages as the implementation language. According to Martin [1985] they require less human effort, less skill and less time, because they are easy to use, developer friendly and have automatic features. Furthermore, the amount of rework is usually smaller, because the code includes less errors and applications are easy and quick to change.

Even though with application generators it has become possible for end users to create their own applications, in this study we study exclusively the productivity of application generators for DP professionals. According to several case reports using these tools DP professionals have been able to improve significantly productivity of application development [Holtz, 1979; Read & Harmon, 1981; Sumner & Benson, 1988]. A few experiments have been conducted in order to find out how productive these application generators really are. The findings vary noticeably showing little evidence in support of Martin's [1985] claims of order of magnitude productivity improvements. However, when these experiments were studied more closely, we found out that they differed substantially in their research settings.

In this paper we present a framework of the main factors having an impact on the productivity. We review the productivity experiments presented in the literature in the light of our framework.

In addition we conducted empirical experiments evaluating the combined effect of application generator and development strategy on productivity.

2. MEASURE OF PRODUCTIVITY

Productivity means the amount of goods or services that can be produced for a given input of labor or expense [Jones, 1986c]. In literature productivity of application development is regarded as resulting application divided by work effort. Although quality is a very important aspect of the product, in the productivity experiments the work product covers only the size and complexity of the application. Furthermore, despite the fact that the maintenance work requires half [Muukari et al, 1983] or even 80% [Martin, 1985] of the DP-resources, none of the productivity studies covers maintenance phase; they either study development life cycle from the preliminary study to acceptance of the product or only limited part of the life cycle - the programming and testing phase. In order to make the difference between these two perspectives clear, we call the former development and the latter programming.

2.1 Measure of Product

To count the number of application code lines produced is the most frequently used measure of product size. It was used in all productivity experiments presented in the literature [Albrecht, 1979; Boehm et al, 1984; Harel & McLean, 1985; Kuvaja, 1988; Misra & Jalics, 1988; Verner & Tate, 1988] except the studies of Rudolph [1983a, 1983b]. The advantage of this measure is its simplicity and cost effectiveness. But, due to the large variety of ways to count the lines the results can sometimes differ even 500% [Jones, 1986c]. Thus, if the way of counting lines is not known, the method is not suitable for comparing results of different experiments. Nor is it suitable for comparing productivity of different generations of programming languages, because it penalizes high-quality languages and often moves in the wrong direction as productivity increases [Jones, 1986b]. In addition it has been noted that good programmers write more structured and more compact code in less time than poor programmers [Jeffery & Lawrence, 1979]. Thus lines of code metrics fails to show the productivity difference between programmers. Still another disadvantage is that the complexity of the program cannot be taken into account with lines of code [Jones, 1986c].

Function point analysis (FPA) is a method of characterizing the size and complexity of applications. It was used in the studies of Rudolph [1983a, 1983b], Albrecht [1979], Kuvaja [1988] and Verner & Tate [1988]. The product is measured by counting the number of external user inputs, outputs, inquiries and master files delivered by the development project. These counts are weighted by numbers designed to reflect the function value to the customer and the complexity level [Albrect, 1979; Albrecht & Gaffney, 1983]. The sum of weighted counts give the total of raw function points. Raw function points are finally adjusted by the level of influence of 14 adjustment factors. This adjustment can modify the raw function points within the range +- 35%.

Analysis is based on user's external view of a program rather than inspection of source code. The weak point in function point analysis is that it underweights applications of high complexity [Symons, 1988]. Another disadvantage is that function points cannot be counted automatically; own judgement has to be used in determining the complexity and number of functions. This gives room for subjectivity and error. Rudolph [1983a] claimed that the upper limit of error range is +- 30%. in addition he stated that the error is less than +- 10 % when the application measured is in operation or has detailed design specifications [Rudolph, 1983a].

Function point analysis is considered independent of the programming language [Rudolph, 1983a] and development methodology [Navlakha, 1986]. Although technology factors have a slight effect on the function point measure, it can be used for comparing productivity effect of different factors and different generations of programming languages.

2.2 Measure of effort

Effort is measured as 'time spent' rather than using "monetary units that are unequal between different organizations and even more unequal between different countries [Rudolph, 1983a]". Furthermore, in many situations it is the time taken, which is more critical than the financial costs of the application development [Rudolph, 1983a]. In literature most productivity experiments measured effort in work hours recorded by the programmers during the development task.

2.3 Selected measure of productivity

Because we study the effects of different factors on application development, we measure work product in function points covering size and complexity of the product. As the application development is highly labor intensive and more often it is the human resources which are the limiting element in DP work, we measure only work effort. Work hours are used to overcome the difficulties of monetary units.

The results of those studies, which used only lines of code metrics [Boehm et al, 1984; Harel & McLean, 1985; Misra & Jalics, 1988], were transformed to function points. This was made with the help of Jones [1986c] notion of the approximate number of source statements required to code one function point in different languages (Cobol - 106 LOC/FPA, Pascal - 91 LOC/FPA).

3. FRAMEWORK FOR PRODUCTIVITY FACTORS

In order to being able to improve the productivity it is essential to recognize factors affecting it. And, indeed, several factors influencing the productivity of application development process have been specified. Jones has made one of the largest surveys listing 45 productivity factors [Jones, 1986c]. According to Jones these factors "account for more than 90% of the real variations in productivity that occur in industry". Benbasat and Vessey [1980] presented a list of factors influencing programming productivity. It was expanded from Chrysler's framework [1978]. In addition to these lists, the COCOMO-model by Boehm [1981] specifies most important productivity factors.

Our framework for analyzing the effect of certain factors on productivity is presented in figure 1. The framework is based on the previous frameworks presented in the literature. The factors included are the ones that have been stated as having a great importance and that also have been analyzed in the productivity experiments.

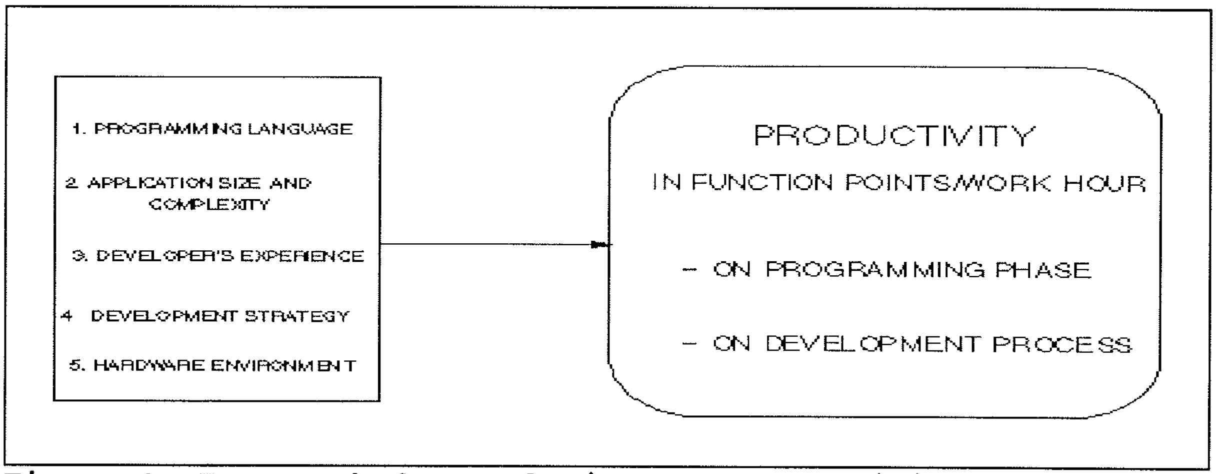


Figure 1. Framework for analyzing the productivity factors.

The factors are the following:

* Programming language

Languages are divided into three categories: traditional third generation languages (3GLs -e.g. COBOL, FORTRAN, PL/1), modern third generation environments (3½GLs - e.g. JSP-COBOL, FORTRAN with SQL and ISPF) and application generators (4GLs -e.g. Linc, Ingres, Clarion, Focus).

* Size and complexity of the application

Different measures are analyzed according to their ability to measure productivity differences. The effects of size and complexity are examined by dividing the applications into three size categories as measured in function points (FPAs).

* Experience

Experience is analyzed by paying attention to experience in programming language (tool), in hardware environment, in application area and in structured programming.

* Development strategy

The projects are divided according to the application development strategy. They are categorized and named according to Davis [1982] to the main categories of linear projects and prototyping projects.

* Hardware environment

The impact of hardware environment is studied by dividing hardware to mainframes, minicomputers and microcomputers.

3.1 Programming language

Increased productivity and ease of use are characteristics of higher generation of programming languages. The general purpose of every new language is to offer a more than decade leap in productivity compared to the previous generation [Martin, 1985].

The first generation of computer languages was machine language. The program had to be written with instruction set of the machine and using physical addresses. The second generation of languages came into use in the mid 50's. They were using symbolic addresses rather than physical machine addresses. The languages of the third generation, which came into use in the 60's, are called high-level languages. With the third generation, the language became more portable. Because of interpreters and compilers, programmer could for the first time code programs using English words and mathematical notations [Martin, 1985].

Fourth generation languages were introduced in the end of 70's. They permit applications to be generated with fewer lines of code and smaller work effort than would be needed with third generation languages like COBOL, PL/1, and FORTRAN. The maintenance costs are expected to decrease, because changes to the applications are easy and quick to make. In addition to employing sequential statements like third-generation languages, they employ a diversity of other mechanisms, such as filling in forms and panels, screen interaction and computer-aided graphics. [Martin, 1985]

Application generators are in general defined as interactive and integrated software systems aiming at improving information systems development and maintenance by automating parts of the system development life cycle [Kuvaja, 1988]. They consist of number of features such as a screen generator, report generator, data base management system and nonprocedural language. Yet, they vary in their power and capabilities. In this paper we study only those application generators, which enable creation of an entire application.

With 3½GL we refer to modern third generation environments, which can take advantage of one or more of the 4GL-features. The main functional difference between 3½GLs and application generators is - as Kuvaja states [1988] - the integrity of different functions. In application generators all functions are integrated: this gives them powerful new possibilities such as prototyping and effective application modification aids. This differentiates them from the modern third generation development environments.

Various articles and case-studies report significant improvements gained with application generators. Martin [1985] argued that productivity of development and maintenance can be even 10 - 80 times better than with traditional languages. Typically 2 - 60 times better productivity have been reported [Xephon, 1984; Holtz, 1979; Read & Harmon, 1981; Sumner & Benson, 1988]. The research studies report productivity to be ½ - 400 times the productivity of third generation languages.

3.2 Application size and complexity

Development of large systems is in general more expensive and more time consuming than development of small systems. It is difficult, though, to differentiate the productivity effect of size from other factors. Usually as the application grows in size, also its processing gets more complex, the number of interfaces and the need for communication between project participants increases. According to Jones [1986c] the effort used to activities like managing, documenting and planning grows quicker than effort used to other activities; thus the percentage of non-productive work increases in development process causing the productivity to fall down.

Experiments studying the effect of application size on productivity have been carried out mainly with third generation languages. It is agreed, that generally it does take shorter time to program a small application [Jeffery & Lawrence, 1979; Vessey, 1986; Albrecht, 1979; Behrens, 1983; Chrysler, 1978]. One study [Chrysler, 1978] found the number of input files, control breaks and totals, input edits and input fields to have a statistically significant impact on productivity, and another study [Jeffery & Lawrence, 1979] found no evidence of this. Some results show that productivity increases as the number of code lines increases [Vessey, 1986; Jeffery & Lawrence, 1979]. On the other hand the productivity has been found to decrease as the size in function points increases [Albrecht, 1979; Behrens, 1983]. Differences in the measures may explain some of the inconsistency of the results. Jeffery and Lawrence [1979] suggested another explanation to these confusing results: The relationship between application size and productivity is more complex than expected. It might be approximately linear over small intervals only.

The fact that application development process always includes some fixed overhead effort, might explain differences in productivity. In the range of small applications the total effort may consist of mostly this overhead causing low productivity results. When the application grows in size, the overhead is spread over a greater work product and the productivity increases. But, as the application grows even more, more non-productive work is associated with the process resulting

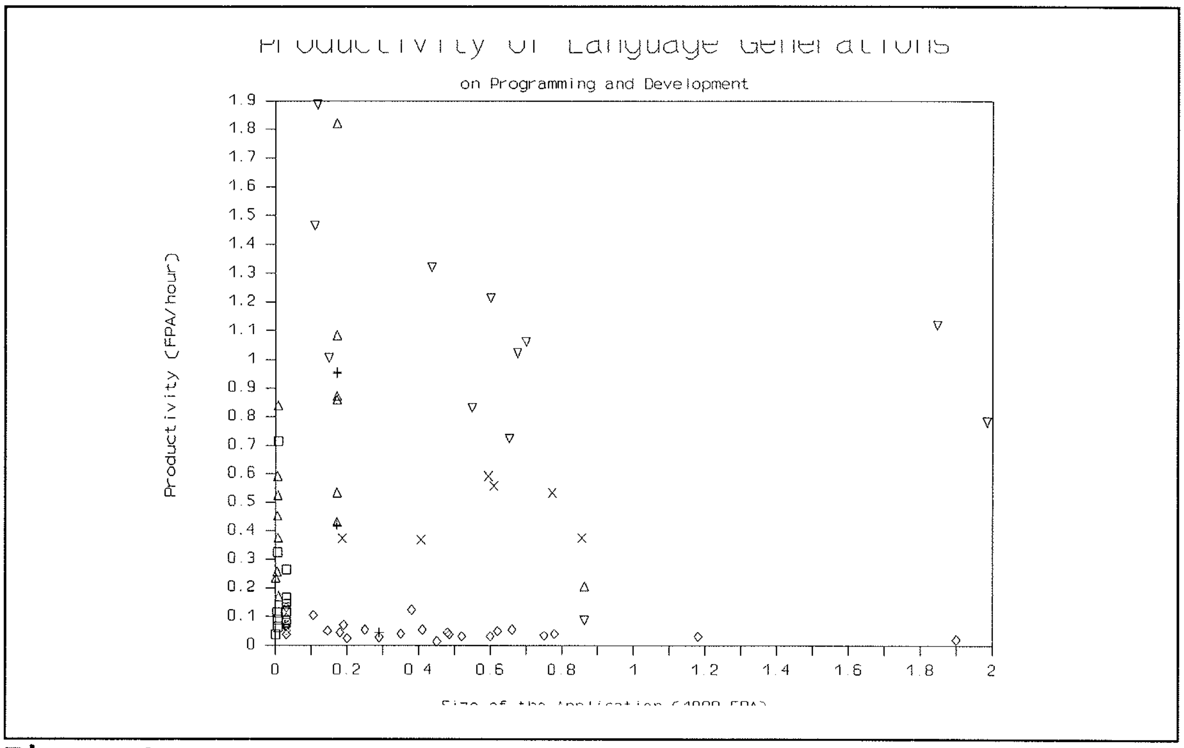


Figure 2.

where

▼ developed with 4GL ▲ programmed with 4GL
X developed with 3½GL + programmed with 3½GL
♦ developed with 3GL ■ programmed with 3GL

One application developed with 3½GL was left out of the picture because of its substantially large size (4770 FPA - productivity 0.28 FPA/hour) in comparison to other applications. This was done purely for the sake of illustration.

[Rudolph, 1983a, 1983b; Albrecht, 1979; Boehm et al, 1984; Harel & McLean, 1985; Kuvaja, 1988; Verner & Tate, 1988; Misra & Jalics, 1988].

in decreasing productivity.

The figure 2. shows the productivity results of different language generations (3GL, 3½GL and 4GL) in respect to the size of the application. Both programming productivity and development productivity can be observed from the figure. The productivity of application generators (4GLs) is better than that of traditional third generation languages. The difference between 3½GL and 4GL is not so clear-cut, but in general 4GLs have been more productive, especially when the productivity of development is examined.

3.2.1 Application size categories

Jones [1986a] divided applications into eight size ranges. The size ranges were measured in lines of code. Even though it has not actually been stated by Jones, it has been assumed that the code lines refer to Cobol code [Kuvaja, 1988]. Thus we calculated corresponding size ranges in

function points using the ratio of 106 Cobol code lines per 1 function point.
Because of the small number of experiments (8 experiments, 71 applications) and the fact that there were no studies about large or super large applications, we were compelled to use a condensed categorization as follows

0 - 20 FPA	small
20 - 300 FPA	medium
300 - 4800 FPA	large

The following statements are based on Jones' [1986a] claims of specific tools and methods giving the best productivity depending on the size range. In the category of 0 - 20 function points the best productivity have been reached with spreadsheet languages and data base query languages. In the category of 20 - 300 function points the most productive way to develop programs is to use application generators, automated design tools and good programming languages.

When the application gets larger than 300 function points, the best results can be achieved with a combination of good requirements, design, prototyping and defect removal. Usually application generators reduce the need of human resources at the expense of hardware resources. If application generators are used for systems with high transaction volumes or complexity, careful attention must be paid to machine performance. Most 4GLs are not designed for heavy-duty computing [Martin, 1985]. Thus, as the application gets very large and the performance constraints become more severe, languages closer to machine language may be better choice than 4GLs. Jones [1986a] recommends utilization of reusable code modules and libraries of standard design.

The research experiments show rather low productivity with 3½GLs and 4GLs when the application size is small. When the size grows a bit, the productivity increases. But it begins to decrease again as the application grows larger. This phenomenon is in accordance with our assumption of the effects of overhead effort associated with size and complexity on productivity.

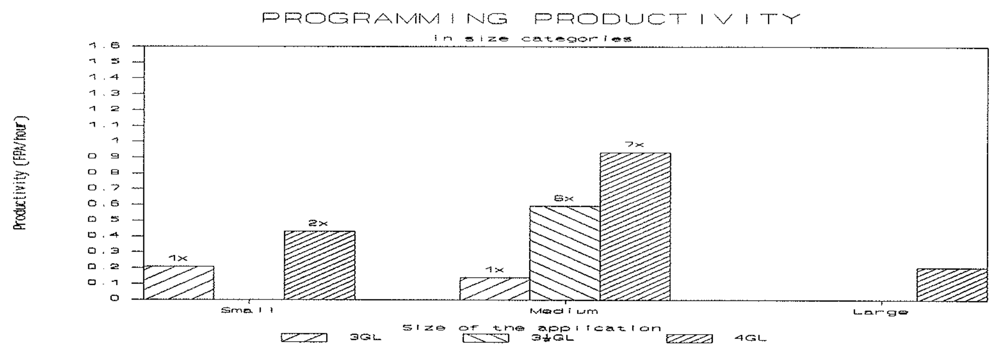


Figure 3. . Harel & McLean, 1985; Boehm et al, 1984; Misra & Jalics, 1988; Kuvaja, 1988; Verner & Tate, 1988.

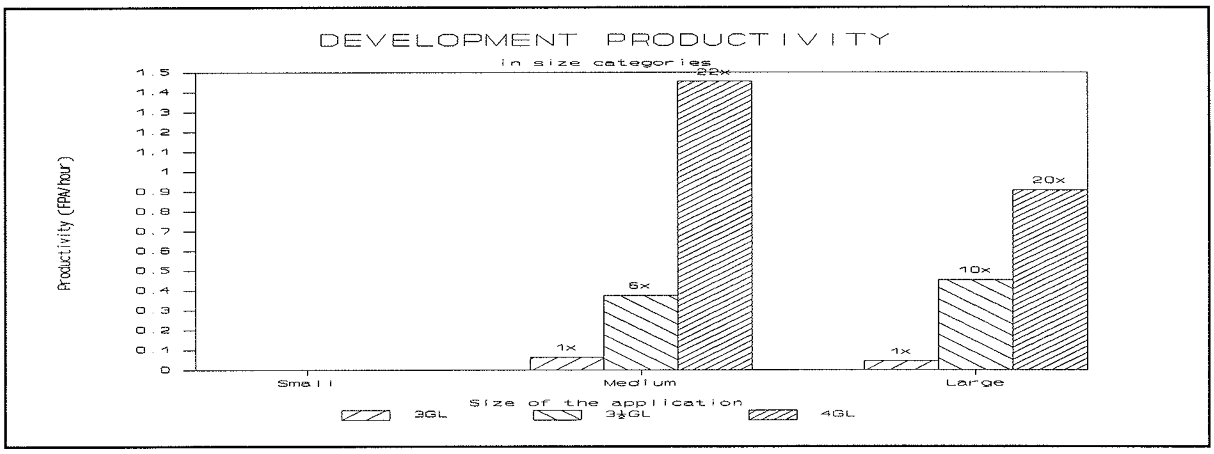


Figure 4. Boehm et al, 1984; Rudolph, 1983a, 1983b; Albrecht, 1979; Verner & Tate, 1988.

If the effect of application size and complexity is examined more closely by diving the applications into size categories, this phenomenon is easily observed (Figures 3 and 4). The productivity ratio between 4GL and 3GL is as well in programming as in developing at its best when the application is medium sized (20 - 300 FPA). It also shows that the productivity of application generators will decrease noticeably when the application gets large. But, we would like to point out the differences between productivity of programming and development; in the range of 20 - 300 FPA the productivity ratio between 3½GL and 3GL seem to be the same, 6:1, regardless of the life cycle phases examined. But with application generators the productivity is even better when also preliminary work and design has been studied. These development productivity results are based mainly on the findings of Rudolph [1983a]. The results of Rudolph's studies are remarkably better in comparison to the results of other studies. Rudolph attributes this to programmers' experience in the problem area and small project groups. Unfortunately there is not enough information about other factors, such as development strategy, for analyzing their impact on these results.

In addition, the productivity of 3½GL doesn't seem to fall down as the application size increases. On the contrary, the ratio to traditional languages improves further. The results support Jones's statement about tools giving the best productivity in different size categories. Unfortunately there are no results of the productivity of application generators in the size range of 20 - 75 FPA. This is the category in which - according to Jones - application generators have proven to be the winners in productivity. Anyway, application generators tend to be at their best in medium sized applications, but when the size increases even further, modern third generation environments challenge their superiority.

3.3 Experience of developers

According to Boehm [1981] the ability and selection of the people assigned to the project have a great impact on application development process.

Productivity impacts of experience, has been studied by several researchers. Again the results are contradictory: Chrysler [1978] found that programmer's experience improves productivity, as did Cheney [1984] and to some extent also Vessey [1986]. But Behrens [1983] and Jeffery and Lawrence [1979, 1985] could not find such a relationship. In any case, it seems to be difficult to examine and quantify human factors. The personal characteristics cannot be totally eliminated; they as well as the different measures applied obscure the research findings.

Jones divided [1986c] programmer's experience to experience in programming language (tool), problem area and structured programming. A programmer experienced in all these subcategories is about 20 times more productive than inexperienced programmer. The difference between the best- and worst-case situation would be even more extreme. Martin [1985] claims that a good programmer would easily achieve 10 times the productivity of poor programmer with any language, but with 4GLs the difference might rise to 100 to 1. In addition it is easier to achieve a higher level of skill with 4GL than with conventional languages, notes Martin [1985].

Even though effect of experience has been noted almost in every experiment, it is not explicitly reported. Therefore we have gathered all information on the experience found in the reports of productivity studies into figure 5. Based on the results conclusions of the impact of experience on productivity are difficult to make. As the information about experience in application area, hardware environment and structured programming is insufficient for comparison of their effect on productivity, we analyze exclusively the effect of experience in programming language on programming productivity. The results give an impression that in the category of small applications application generators would have more effect on the productivity of novices than that of experts. Application generators seem to be most profitable when experienced programmer is using them in programming of a medium sized application.

Figure 5. PROGRAMMERS' EXPERIENCE IN THE EXPERIMENTS

EXPERIENCE EXPERIMENT	IN LANGUAGE	IN HARDWARE ENVIRONMENT	IN PROBLEM AREA	IN STRUCTURED PROGRAMMING
RUDOLPH 1983A	The tool been in use at least 1 year. The experience of programmers not mentioned.	Not mentioned	Mean 6 years. Each team had at least 1 member with >= 8 years.	0 - 8 years. Mean 3.4 years.
RUDOLPH 1983B	Not mentioned	Not mentioned	Not mentioned	"Developed by highly professional DP organizations, although several systems were implemented by relative junior staff member" [Rudolph, 1983b].
ALBRECHT 1979	Not mentioned	Not mentioned	Not mentioned	Developed by IBM's DP Service organization. experience not mentioned.
BOEHM, GRAY, SEEWALDT 1984	Group Average 1. 1 month 2. 17 months 3. 7 months 4. 3 months 5. 30 months 6. 16 months 7. 9 months	Group Average 1. 0 months 2. 1 months 3. 12 months 4. 5 months 6. 3 months 7. 4 months 8. 0 months	Not mentioned	Group Average 1. 25 months 2. 47 months 3. 42 months 4. 30 months 5. 54 months 6. 46 months 7. 60 months
HAREL & MCLEAN 1985	Beginners had written 5 - 20 programs, experts more than 20 programs.	Not mentioned	All had at least six months exposure to the application environment.	Not mentioned
KUVAJA 1988	Novices were inexperienced Professionals had used the tool in question for more than one year.	Familiar working environment for experienced. If unfamiliar, some time (not recorded) was spent to get acquainted with it.	All inexperienced. All developers were acquainted with the test application documentation. Time spent not recorded.	Professionals had worked as systems developers for years. DP students from 4-5 studying year were used as inexperienced. No other information given.
VERNER & TATE 1988	Language was unfamiliar.	Not mentioned	The organization had never used computers for administration.	Computing professionals and small group of inexperienced teachers assigned to the process.
MISRA & JALICS 1988	Experience was much lower in one language than in other two languages tested	Not mentioned	Not mentioned	Not mentioned

3.4 Development strategy

Successful project management and control is one of the important factors in application development [Jeffery, 1987]. Every development life cycle consists of preliminary work, design, implementation and installation and maintenance. Depending on the approach employed these main activities are divided into more precise phases.

3.4.1 Linear strategy

Conventional approaches, like Cascade model [Birrel & Ould, 1985], manage the process with the help of rigorous specification and control. The process is carried out in linear fashion: A phase is started only after the previous phase has been completed. Returning to previous phase has to be avoided.

3.4.2 Prototyping strategy

The basic idea of prototyping is to generate a working model of the application instead of writing detailed specifications of it. According to Naumann and Jenkins [1982] prototyping neither requires nor permits prolonged static specifications in development process. The prototype is generated quickly and is demonstrated to the user. Undesirable or missing features identified by the user must be corrected. Several iterations will undoubtedly be required before the application is finished.

3.4.3 Recommendations and praxis

Burns and Dennis [1982] introduced a framework for suitable development strategies depending on complexity and uncertainty of the project. The conventional linear strategy should be used only with projects with high complexity and low uncertainty. When both the complexity and uncertainty are high, a mixed strategy of linear and prototyping approach is more suited. Otherwise prototyping is recommended.

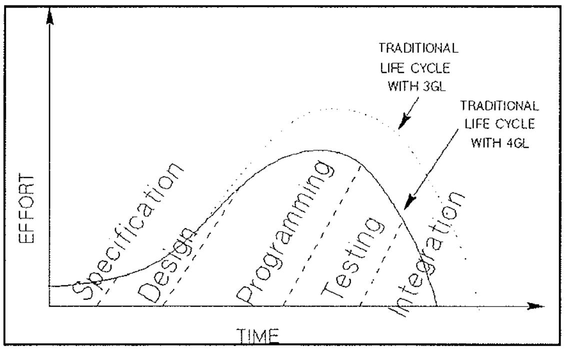

TRADITIONAL LIFE CYCLE WITH 3GL
TRADITIONAL LIFE CYCLE WITH 4GL
EFFORT
Specification
Design
Programming
Testing
Integration
TIME

Figure 6. Use of application generators with linear strategy

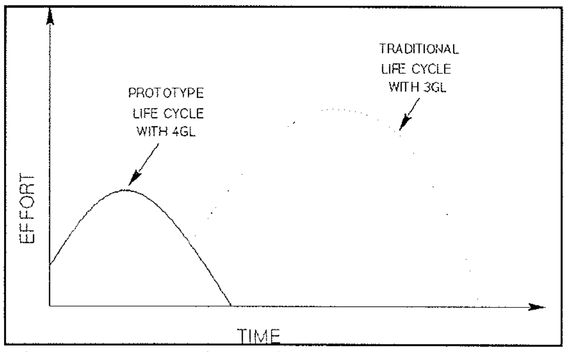

TRADITIONAL LIFE CYCLE WITH 3GL
PROTOTYPE LIFE CYCLE WITH 4GL
EFFORT
TIME

Figure 7. Use of application generators with prototyping strategy.

Martin [1985] illustrates (figure 6.) the productivity improvement gained with fourth generation languages when linear approach is used. Productivity is improved only in programming and testing phase. He argues that if the application generators are to be used most effectively, also the development strategy should be adopted to the technology used. The features of application generators are at their best when prototyping approach is used. The distribution of work effort to different phases changes and the total effort decreases (figure 7.).

Although prototyping is best strategy for 4GLs, most of the research experiments have studied the impact of application generators on productivity only with linear development approach. The figure 8. presents the development strategies used in the experiment. If we compare the implementation of research experiments to Burns's and Dennis's recommendations, we note that most of the experiments had used a development strategy suitable for only the projects with high complexity and low uncertainty. Harel and McLean [1985] stated their test projects to be 'simple or not so simple'. Also the specifications of the test application of Misra and Jalics [1988] gave an impression of low complexity. Thus according to Burns and Dennis prototyping approach should have been used. For the project of Verner & Tate [1988] either protype or mixed strategy

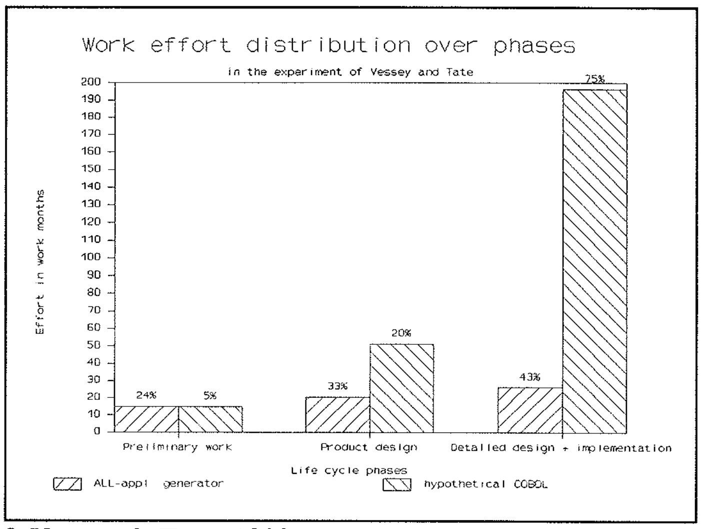
Figure 8. Development methods used in the experiments and the recommendations of Burns and Dennis [1982].

	Linear strategy	**Mixed strategy**
high	-Albrecht, -Harel & Mclean -Kuvaja -Misra & Jalics -Boehm et al	
complexity		
low	**Prototype strategy** - Boehm - Verner & Tate	
Recommended in projects with	low	high
	uncertainty	

is recommended. Due to hypothetical application of Kuvaja's [1988] experiment it is difficult to determine suitable approach. It might have, though, been better to use protyping strategy.

On the other hand prototyping is encouraged only in situations where suitable technology and tools are available. If the program is developed with conventional language, mixed strategy is recommended instead of prototyping [Burns & Dennis, 1982]. Even though the experiment of Boehm, Gray and Seewaldt [1984] was not following this advice, it gives evidence of notable productivity effect of prototyping; protyping groups needed about 45% less time than linear groups in developing the same application with Pascal. The researchers stated: " Prototyping did not tend to produce higher productivity measured in delivered source instructions per man-hour. However, if productivity is measured in equivalent user satisfaction per man-hour, protyping did tend to be superior [Boehm et al, 1984]".

If prototyping approach can improve noticeably the productivity of conventional languages, the productivity improvement would undoubtedly be even more outstanding with application generators. In Verner's and Tate's experiment a group of programmers inexperienced in application generator used, developed a large application (863 FPA) with prototyping approach. The productivity was compared to a hypothetical productivity of the project implemented with Cobol: The productivity ratio was estimated to be about 4:1.



9 Verner & Tate, 1988

The figure 9. illustrates how prototyping changed the distribution of work effort over the development life cycle phases. Use of application generator together with prototyping reduced the work effort mainly in implementation but also in design phase. Thus in order to study the real productivity effects of application generators, examination of programming productivity is inadequate; attention should be paid to development strategy and to the whole application development life cycle from preliminary study to completion of the application or even to maintenance phase. In addition to the study of Verner & Tate [1988] only Rudolph has examined development productivity of application generators [1983a]. All other studies of application generators have examined only programming productivity and they used prespecified applications when, contrary to prototype projects, no productivity improvements in design phase are possible. Moreover, it is possible that the productivity of plain programming and testing phase is better using application generators with prototyping approach rather than with linear approach.

3.5 Hardware environment

The effect of hardware environment on productivity has not received much attention in the studies. It is, however, assumed that productivity improves when the environment gets simpler [Thadhani, 1984]. Thus in microcomputer environment the productivity should be better than in mainframe or minicomputer environments.

Hardware response time does have a certain effect on programmer's productivity. According to Jones response times vary in mainframe environments from two to five seconds, but response times of ten seconds are certainly not rare. The productivity was proven to improve by 60 % (measured in FPAs) only cutting the response time from 2.22 to 0.84 seconds [Lambert, 1984]. Findings of Thadhani [1984] support this result.

Examination of productivity studies doesn't give a clear picture of the effect of hardware environment on achieved productivity. But for example in Kuvaja's study [1988] the productivity of both novice and experienced programmer was better in minicomputer environment than in mainframe environment. Thus it seems that hardware environment does have a small effect on productivity.

3.6 Summary of the framework analysis

In the following figure (figure 10.) we present the productivity experiments in the light of our framework. More detailed information is presented in appendix.

Figure 10. Research settings of productivity experiments

EXPERIMENT		RUDOLPH		ALBRECHT		BOEHM &		ET AL		HAREL		MCLEAN				KUVAJA			MISRA &	JALICS		VERNER&
Applications	11	2	7	16	2	1	1	2	1	3	3	3	3	3	1	1	1	1	2	1	1	1	1
Language GL																							
Application gen.4	X									X		X		X	X		X	X		X	X		X
Modern 3½			X													X			X				
conventional 3		X		X	X	X	X	X	X		X		X									X	
Hardware environment																							
Mainframe	X	X			X	X	X	X	X	X	X	X	X	X		X	X		X				X
Minicomputer			X												X			X					
Microcomputer																				X	X	X	
Experience					2	2	2	2	2														
in language					N	E	N	E	N	N	N	E	E	N	N	N	E	E	E	E	N	E	N
in hardware envir.					N	N	E	N	N								E	E	E				
in problem area	E													N	N	N	N	N	N				
in structured prog.	E				E	E	E	E	E								E	E	E				
Development strategy																							
Linear				X	X	X	X			X	X	X	X	X	X	X	X	X	X	X	X	X	
Prototype								X	X														X
Measurements																							
Output																							
FPA	X	X	X	X										X	X	X	X	X	X				X
Lines of code				X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
Input																							
Work month																							X
Work hour	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	
Size																							
Small 0-20FPA										3	3	3	3							1	1	1	
Medium 20-300FPA	3	2	1	5	2	1	1	2	1					3	1	1	1	1	2				
Large 300-4800FPA	8		6	14																			1
Life cycle																							
Preliminary work	Y	Y	Y	Y	Y	Y	Y	Y	Y														Y
Design	Y	Y	Y	Y	Y	Y	Y	Y	Y														Y
Implementation	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

N	Novice
E	Experienced
2	experienced >= 1 years experience

On the basis of the literature analysis made, it is quite obvious, that there is still a lot of research to be done in order to get more complete picture of the productivity effects of application generators on DP work. Furthermore it is evident, that we should not forget to take into consideration also the coverage of the application life cycle as well as the factors affecting the productivity, when any comparisons between different productivity results are made.

In our opinion especially the development strategy has been overlooked in the productivity studies of application generators. Therefore we conducted experiments, where we varied the development strategy keeping the other factors of the framework constant.

4. EMPIRICAL EXPERIMENTS

Our purpose was to examine the impact of combined effect of application generator and development strategy on productivity, because most of the productivity experiments have studied application development with linear development strategy only and because application generators are expected to be at their best with prototyping approach.

4.1 Research settings

We carried out two empirical experiments (Figure 11.): a linear project and a prototype project.

4.1.1 Linear project

In the linear project a novice programmer programmed a hypothetical application with application generator. A prespecified test application was used. The specifications were the same that were used in Kuvaja's [1988] experiment. Thus we could calibrate our measuring methods to measures of Kuvaja. In addition, because Kuvaja's experiments were conducted in mainframe or minicomputer environment and we used microcomputer environment, we could examine, how the hardware environment affects the programming productivity of novice programmer.

Project	P	L
Applications	2	1
Language GL		
Application gen. 4	X	x
Modern 3½		
conventional 3		
Hardware environment		
Mainframe		
Minicomputer		
Microcomputer	X	X
Experience		
in language	N	N
in hardware env.	E	E
in problem area	N	N
in structured prog.	N	N
Development method		
Linear		**X**
Prototype	**X**	
Measurements		
Output		
FPA	X	X
Lines of code		
Input		
Work month		
Work hour	X	X
Size		
Small 0-20FPA		
Medium 20-300FPA	2	1
Large 300-4800FPA		
Life cycle		
Preliminary work	Y	
Design	Y	
Implementation	Y	Y

where
P Prototype project
L Linear project

and

N Novice
E Experienced
2 experienced >=

Figure 11. Research setting of our experiments.

4.1.2 Prototyping project

In the prototyping project two versions of an application were created according to needs of the Society of Forestry in Finland. The application had a low complexity, but had some uncertainty associated with it. This was, according to the recommendations of Burns and Dennis [1982], an ideal project for prototyping. In addition, the application was medium sized - a size category in which application generators have shown biggest productivity.

4.1.3 Differences

The biggest difference to the linear project was that in prototype project no formal written specifications were made - the developer designed the application during programming. Another important difference was that in prototype project the same person performed all activities from the preliminary work to completion of the application, where as in linear project the programmer only programmed and tested an application to meet the requirements made by another person(s).

The application generators were:

- in the linear project
  - * Clarion Professional Developer
- in the prototype project
  - * Ingres PC-version 5/02a
  - * Paradox PC Version 2.01.

Otherwise the research settings were similar in both experiments: microcomputers were used as development environment, both applications were medium sized on-line and real-time business systems and the developers were experienced in hardware environment but unskilled in programming language, application area and structured programming. The programmer of the linear project was one of the developers of the prototyping project as well. Thus personal characteristics should not affect the productivity difference between the two applications she produced. The experience of the programmers is presented in the following figure 12.

Figure 12. EXPERIENCE IN OUR EXPERIMENTS

EXPERIENCE EXPERIMENT	IN THE LANGUAGE	IN THE HARDWARE ENVIRONMENT	IN THE PROBLEM AREA	IN STRUCTURED PROGRAMMING
Linear Project	The programmer had no former experience in the tool. Before experiment she first looked through tutorial and some examples and then developed a small program by herself. - Novice	Programmer had used micro-computers several years and was using them at work daily. - Expert	No experience on this problem area. - Novice	The programmer was 4th year student of Business Information Systems at Helsinki School of Economics. Thus theory familiar, but little experience. - Novice
Prototype project	Both developers had developed one medium sized application with the application generator used in the experiment. - Novices	Both developers had used microcomputers several years and were using them daily at work. - Experts	The developers had developed together an application of the same type using another application generator. No other experience. - Novices	Both were 4th year students of Business information Systems at Helsinki School of Economics. Thus theory familiar but little experience. - Novices

4.2 Measurements

The developers measured the size and complexity of the applications with function points. The application of the linear project was finally measured to have 171.8 corrected function points, which was very close to the mean value - 171.3 FPA - of the estimates given by the programmers in Kuvaja's experiment. Thus we confirmed that our way of counting function points (we noticed that the function point estimate varied easily +-30%) doesn't affect comparison results between our and Kuvaja's experiment. Using the same way of counting application of the prototype project was measured to have 133.6 corrected function points.

The work hours were recorded by the programmers during the test. In the first experiment the unit of measure was 5 minutes. In the second experiment the programming and testing effort was recorded in minutes, the work needed to other activities was estimated in hours. Details are reported in figure 13.

Figure 13. Work effort distribution over life cycle phases on empirical experiments

Life Cycle Phases	WORK EFFORT DISTRIBUTION OVER PHASES Linear Project Clarion	Prototype project Ingres	Paradox
Preliminary work	Unknown, because we created a hypothetical application according to specifications made in the Institute of Information Processing Science, University of Oulu, Finland.	Both developers spent about 20 hour. In addition the representatives of the association and the supervisor of the project spent about 12 hours in meetings and other activities associated with the project.	
Design	unknown. The specifications included processing logic, information types and databases and screen and report layouts. Programmer spent about 4 hours to get familiar with the specifications.	27 hours	35.5 hours
Programming & Testing	169 hours, which includes all activities after getting acquainted with specifications.	24 hours	14 hours

4.3 Results

We measured programming productivity in both experiments. In the **linear** project the programming productivity was **1.02 function points per hour** (including some design work, programming and testing). However, in the **prototype** project the programming productivity was **2.62** (Ingres) and **2.71** (Paradox) **FPA/hour** (including all design work, programming and testing). In prototype project we could also measure development productivity (productivity of all phases from preliminary work to completion of the application): with Ingres development productivity was 1.61 and with Paradox 1.64 function points in an hour. Unfortunately, because we didn't know how much time was spent in preliminary work and design phase in the linear project, we were not able to measure its productivity in development process. Anyway, the productivity of prototype projects were significantly better: In comparison to the linear project the programming productivity was double and the productivity of the whole development life cycle with prototyping exceeds the productivity of programming activities in linear project.

One reason for the much better productivity of prototype project was that with linear project rigorous specifications forced to program the application in a way, that many of the features of the application generators could not be used. On the contrary, in prototype project there were no restrictive specifications and the developers could design and implement the application getting the best of application generator they were using. This benefit was specially noteworthy, since the association had no former experience of the use of computer technology in its administration. Thus the developers were allowed to use their own judgement and creativity in implementation of the application.

Another explanation for the productivity difference was that in linear project it was sometimes difficult for the programmer to interpret the specifications. In addition, the specifications - as all specifications do - included some lapses. Thus a considerably amount of work effort was spent in clarifying the specifications and correcting mistakes. To the contrary, because the developer in prototyping project was assigned to the project from the beginning, she had a better idea of what was to be developed. Moreover, no time had to spend in making or comprehending formal written specifications.

3.2.1 Comparison to previous experiments

The figure 14. compares the results of Kuvaja's and our experiments. The experience of our programmers should be quite similar to the experience of novices in Kuvaja's experiment. All were DP students of 4th - 5th years and were inexperienced in the application generator they were using. Experience in structured programming was about the same. The difference in the experience in the hardware environment is not known. These results show some evidence of better productivity in microcomputer environment than in minicomputer or mainframe environment. More important result is that the productivity in programming phase seem to be better with protyping. The same novice programmer could more than double her productivity by shifting from linear strategy to prototyping. Regardless of experience and hardware environment,

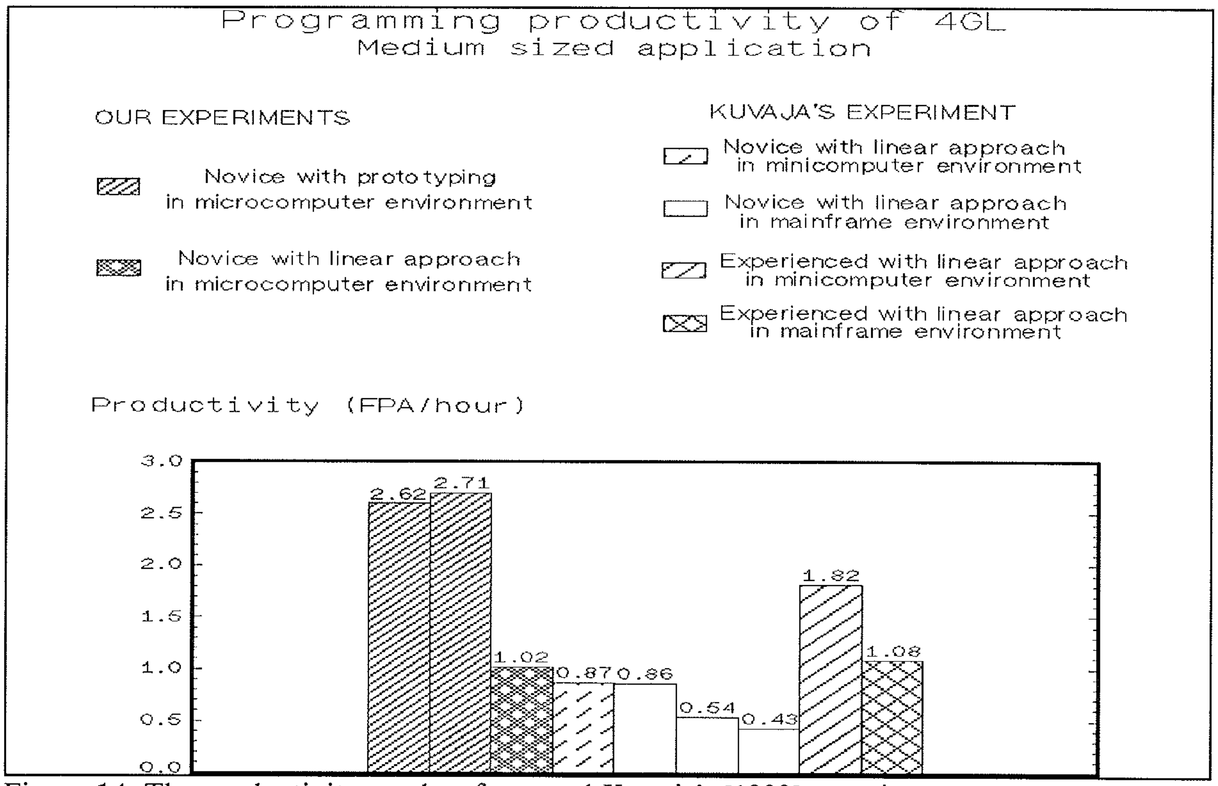


Figure 14. The productivity results of our and Kuvaja's [1988] experiments.

the productivity of application generator is about 2.4 times better with prototyping than with linear approach. And in comparison to traditional way to program applications - Third generation languages with linear approach - the productivity of application generator in microcomputer using prototyping is over 20 times better. In comparison to the novice 3½GL programmer in mainframe environment, the productivity ratio was 6 : 1. These figures concern the category of medium sized applications and analyze only the productivity of programming.

If all the phases from preliminary work to user acceptance of the product are analyzed, our experiment shows about 30 times the productivity of linear project with third generation languages. Furthermore, a productivity ratio of 18:1 was recorded between our prototyping project and a prototyping project carried out by Verner & Tate [1988]. The factors affecting the results were application size and hardware environment: Our application was of medium size and was developed in microcomputer environment, where as the application of Verner & Tate was large and was created in mainframe environment.

5. CONCLUSIONS

The results of this study supports James Martin's suggestion of productivity gains of 10:1 to 80:1 with application generators in comparison of third generation languages. Improvement of 30 to 1 in development process (i.e. preliminary work, design, programming and testing) was achieved by novice programmers using application generators with prototyping. As the productivity ratio in the range of medium sized applications is expected to be better for experts than for novices, the improvement would probably be even more outstanding, if the level of experience would have been higher. In addition, the results support the notion of application generators being most productive in medium sized projects, and the notion of simple and developer-friendly hardware environment having a positive effect on productivity.

Furthermore, an important aspect in favor of prototyping approach is that in prototyping project the programmers were more satisfied with working procedures and conduction of the project than with traditional linear approach. In prototype project the task of a programmer is more motivating and versatile allowing the programmer to use her/his own judgement and creativity in design and implementation of the application. The leverage effects can be remarkable and widespread. Thus the increased job satisfaction with prototyping approach is an issue that should by no means be neglected.

Finally, on the basis of this study, it is obvious, that in order to examine the real productivity effects of application generators, a formal experiment using linear strategy and prespecified test applications is inadequate. All the productivity effects of the tool during the whole application life cycle from the preliminary work to implementation (or to maintenance) should be observed and a development strategy suiting the project should be used. In many cases this would mean prototyping with application generators - a combination, offering powerful possibilities for significant productivity improvements.

6. LIMITATIONS OF THE STUDY

From the generalization point of view the greatest lack of this study is the small number of experiments. Thus personal characteristics of the developers may bias the results.

7. ACKNOWLEDGEMENTS

This paper is based on my master thesis, and a seminar report written together with Johanna Pajunen. I am very grateful to her and the representatives of the Society of Forestry in Finland for making the experiments possible. I would also like to thank Ph.Lic. Pasi Kuvaja for being willing to help me. In addition, I would like to give my warmest thanks to Econ.Lic. Timo Saarinen and MBA Jukka Heikkilä for support and guidance.

REFERENCES

Albrecht A.J., "Measuring Application Development Productivity", Proceedings of the Joint SHARE/GUIDE/IBM Application Development Symposium, Oct **1979**, 83 - 92.

Albrecht A.J. & Gaffney J.E., "Software Function, Source Lines of Code, and Development Effort Prediction: A Software Science Validation", IEEE Transactions on Software Engineering, SE-9, 6, **1983**, 639-648.

Alloway R.M. & Quillard J.A., "User Managers' System needs", MIS Quarterly, 7,2, **1983**, 27-41.

Behrens C.A., "Measuring the Productivity of Computer Systems Development Activities with Function Points", IEEE Transactions on Software Engineering, SE-9, 6, **1983**, 648 - 652.

Benbasat I. & Vessey I., "Programming and Analyst Time/Cost Estimation", **MIS Quarterly** 4, 2, **1980**, 31 - 43.

Birrell N.D. & Ould M.A., "A Practical Handbook for Software Development", Cambridge University Press, **1985**.

Boar B.H., "Application Prototyping", John Wiley and Sons, **1984**.

Boehm B.W., "Software Engineering Economics", Prentice-Hall, Inc., Englewood Cliffs, New Jersey, **1981**.

Boehm B.W., Gray T.E., Seewaldt T., "Protyping vs. Specifying: A Multiproject Experiment", IEEE Transactions on Software Engineering, SE-10, 3, May **1984**, 290 - 303.

Boehm B.W. & Papaccio P.N., "Understanding and Controlling Software Costs", IEEE Transactions on Software Engineering, 14, 10, October **1988**, 1462 - 1476.

Burns R.N. & Dennis A.R., "Selecting the Appropriate Application Development Methodology", Data Base, Fall **1985**, 19 - 22.

Cheney P.H., "Effects of Individual Characteristics, Organization Factors and Task Characteristics on Computer Programmer Productivity and Job Satisfaction", Information & Management, 7, **1984**, 204 - 214.

Chrysler E., "Some Basic Determinants of Computer Programming Productivity", Communications of the ACM, 21, 6, **1978**, 472 - 483.

Davis G.B. ,"Strategies for information requirements determination", IBM Systems Journal, 21, 1, **1982**, 4-29.

Harel E.C. & McLean E.R., "The Effects of Using a Nonprocedural Computer Language on Programmer Productivity", MIS Quarterly, June **1985**, 109 - 120.

Holtz H., "A Nonprocedural Language for Online Applications", Datamation, April **1979**, 167 - 176.

Jeffery D.R., "Software Engineering Productivity Models for Management Information System Development", Critical Issues in Information Systems Research, (ed.) Boland R.J.Jr. & Hirschheim R.A., John Wiley & Sons Ltd., **1987**, 113 - 134.

Jeffery D.R. & Lawrence M.J., "An Inter-Organizational Comparison of Programming Productivity", Proceedings of 4th International Conference on Software Engineering, **1979**, 369 - 377.

Jeffery D.R. & Lawrence M.J., "Managing Programming Productivity", The Journal of Systems and Software, 6, 1, **1985**, 49 - 58.

Jones T.C., "Appropriate Tools, Methodologies Ease Development Effort", Computerworld Feb 24, **1986a**, 48 - 49.

Jones T.C., "How Not to Measure Programming Productivity", Computerworld Jan 13, **1986b**, 65 - 76.

Jones T.C., "Programming Productivity", New York, **1986c**.

Kuvaja P., "An Experimental Analysis of the Selection and Productivity of Application Generators for Professional Use", Research papers series A 10, University of Oulu, Aug **1988**.

Lambert G.N., "A Comparative Study of System Response Time on Program Developer Productivity", IBM Systems Journal, vol 23, no 1, **1984**, 36 - 43.

Martin J., "Fourth-Generation Languages, Volume 1 Principles", Prentice-Hall, Inc., Englewood Cliffs, New Jersey, **1985**.

Misra S.K. & Jalics P.J., "Third-Generation versus Fourth-Generation Software Development", IEEE Software, 5, July **1988**, 8 - 14.

Muukari M., Saarinen T., Sääksjärvi M., "Osallistuminen tietosysteemien kehittämiseen, menetelmät, välineet ja tilanne Suomessa", Research report 1/83, Tietotekniikan kehittämiskeskus Ry., **1983** (in Finnish).

Naumann J D. & Jenkins M.A. "Prototyping: The New Paradigm for Systems Development", MIS Quarterly, September **1982**, 29 - 44.

Navlakha J., "Software Productivity metrics: Some Candidates and Their Evaluation", AFIPS Conference Proceedings, june 16-19, vol 55, AFIPS Press, Las Vegas Nevada, **1986**.

Read N.S. & Harmon D.L., "Assuring MIS success", Datamation, Volume 27, Number 2, Feb **1981**, 109 - 120.

Rudolph E.E., "Productivity in Computer Application Development", University of Auckland, New Zealand, March **1983a**.

Rudolph E.E., "Measuring Software Development Productivity", Proceedings of the Conference of New Zealand Computer Society, 8th of Sept, **1983b**, 417 - 425.

Sumner M. & Benson R., "The Impact of Fourth Generation Languages on Systems Development", Information & Management, 14 Feb **1988**, 81 - 92.

Symons E., "Function Point Analysis: Difficulties and Improvements", IEEE Transactions on Software Engineering, Vol 14, Jan **1988**.

Thadhani A.J., "Factors Affecting Programmer Productivity During Application Development", IBM Systems Journal, vol 23, no 1, **1984**, 19 - 35.

Verner J.& Tate G., "Estimating Size and Effort in Fourth-Generation Development", IEEE Software, 5, Jul **1988**, 15 - 22.

Vessey I., "On Program Development Effort and Productivity", Information & Management, 10, **1986**, 255 - 266.

Xephon Technology Transfer Limited, "Application Generators: Comparative Analysis, User survey, Berkshire, England **1984**.

Appendix Table A1. Research experiments on programming productivity - small applications (under 20 FPA)

SIZE	EXPERIENCE	DEVELOPMENT	HARDWARE	LANGUAGE	EXPERIMENT	SIZE	HOURS		PRODUCTIVITY FPA/HOUR			PRODUCTIVITY RATIO		
CATEGORY	IN LANGUAG	STRATEGY	ENVIRONMENT			FPA			3GL	3½GL	4GL	4GL : 3GL	4GL : 3½GL	3½GL : 3GL
SMALL	NOVICE	LINEAR	MAINFRAME	FOCUS VS.COBOL	HAREL & MCLEAN	(3.61)	31.25	14	(0.12)		(0.26)	2.2 : 1.0		
				FOCUS VS.COBOL	HAREL & MCLEAN	(5.00)	55.5	11	(0.09)		(0.45)	5.0 : 1.0		
				FOCUS VS.COBOL	HAREL & MCLEAN	(6.19)	95	11.75	(0.07)		(0.53)	8.1 : 1.0		
				PRODUCTIVITY					[0.09]		[0.41]	**4.6 : 1.0**		
			MICROCOMPUTE	FOCUS PC	MISRA & JALICS	7.15		19			0.38			
				PRODUCTIVITY					[0.09]		[0.40]	**4.5 : 1.0**		
	EXPERT	LINEAR	MAINFRAME	FOCUS VS.COBOL	HAREL & MCLEAN	(1.02)	26	4.32	(0.04)		(0.24)	6.0 : 1:0		
				FOCUS VS.COBOL	HAREL & MCLEAN	(5.19)	16	8.75	(0.32)		(0.59)	1.8 : 1.0		
				FOCUS VS.COBOL	HAREL & MCLEAN	(8.42)	74	49	(0.11)		(0.17)	1.5 : 1:0		
				PRODUCTIVITY					[0.16]		[0.33]	**2.1 : 1.0**		
			MICROCOMPUTE	DBASE VS.COBOL	MISRA & JALICS	7.15	10	8.5	0.72		0.84	1.2 : 1.0		
				PRODUCTIVITY					[0.30]		[0.46]	**1.5 : 1.0**		
									[0.21]		[0.43]	**2.1 : 1.0**		

The productivity figures in parenthesis have been calculated with the help of Jones notion of how much one function point requires lines of code with different languages.
The size of the test application in the experiment of Misra and Jalics [1988] is calculated from the descriptions given in their report.
The productivity results of Albrecht [1978] are gathered from [Albrecht, 1979; Rudolph, 1983a; Martin, 1985].

Appendix table A2. Research experiment on the programming productivity - Medium sized (20 - 300 FPA) and large applications (300 - 4800 FPA)

SIZE CATEGORY	EXPERIENCE IN LANGUAG	DEVELOPMENT STRATEGY	HARDWARE ENVIRONMENT	LANGUAGE	EXPERIMENT	SIZE FPA	HOURS	PRODUCTIVITY FPA/HOUR			PRODUCTIVITY RATIO		
								3GL	3½GL	4GL	4GL : 3GL	4GL : 3½GL	3½GL : 3GL
MEDIUM	NOVICE	LINEAR	MAINFRAME	PASCAL	BOEHM ET AL.	(31.01)	350	(0.09)					
				PASCAL	BOEHM ET AL.	(31.01)	215	(0.14)					
				PASCAL	BOEHM ET AL.	(31.01)	386	(0.08)					
				FORTRAN	KUVAJA	171.3	407		0.42				
				FOCUS	KUVAJA	171.3	396			0.43			
				CSP	KUVAJA	171.3	320			0.54			
				IDEAL	KUVAJA	171.3	199			0.86			
			PRODUCTIVITY					[0.10]	0.42	0.61	**5.8 : 1.0**	**1.5 : 1.0**	**4.0 : 1.0**
			MINICOMPUTER	INGRES	KUVAJA	171.3	196			0.87			
		PRODUCTIVITY					[0.10]	0.42	0.68		**6.5 : 1.0**	**1.6 : 1.0**	4.0 : 1.0
		PROTOTYPE	MAINFRAME	PASCAL	BOEHM ET AL.	(31.01)	117	(0.27)					
	PRODUCTIVITY							[0.14]	0.42	0.68	**4.7 : 1.0**	**1.6 : 1.0**	**2.9 : 1.0**
	EXPERT	LINEAR	MAINFRAME	PASCAL	BOEHM ET AL.	(31.01)	256	(0.12)					
				COBOL	KUVAJA	171.3	179		0.96				
				JSP-COBOL	KUVAJA	171.3	180		0.95				
				RAMIS II	KUVAJA	171.3	158			1.08			
			PRODUCTIVITY					[0.12]	0.96	1.08	**9.0 : 1.0**	**1.1 : 1.1**	**7.9 : 1.0**
			MINICOMPUTER	MAPPER KIT T.	KUVAJA	171.3	94			1.82			
		PRODUCTIVITY						[0.12]	0.96	1.45	**12.0 : 1.0**	**1.5 : 1.0**	7.9 : 1.0
		PROTOTYPE	MAINFRAME	PASCAL	BOEHM ET AL.	(31.01)	188	(0.16)					
				PASCAL	BOEHM ET AL.	(31.01)	264	(0.12)					
		PRODUCTIVITY							[0.14]				
	PRODUCTIVITY							[0.13]	0.96	1.45	**10.8 : 1.0**	**1.5 : 1.0**	7.1 : 1.0
PRODUCTIVITY								[0.14]	[0.78]	[0.93]	**6.7 : 1.0**	**1.2 : 1.0**	5.5 : 1.0
LARGE	NOVICE	PROTOTYPE	MAINFRAME	ALL	VERNER & TATE	863	4192			0.21			

Appendix table A3. Research experiments on development productivity - Medium and Large applications (20 - 4800 FPA)

SIZE CATEGORY	EXPERIENCE IN LANGUAGE	DEVELOPMENT STRATEGY	HARDWARE ENVIRONMENT	LANGUAGE	RESEARCH	SIZE FPA	WORKING HOURS 3GL	WORKING HOURS 3½GL	WORKING HOURS 4GL	PRODUCTIVITY FPA/hour 3GL	PRODUCTIVITY FPA/hour 3½GL	PRODUCTIVITY FPA/hour 4GL
MEDIUM	NOVICE	LINEAR	MAINFRAME	PASCAL	BOEHM ET AL.	(31.01)	589			(0.05)		
				PASCAL	BOEHM ET AL.	(31.01)	459			(0.07)		
				PASCAL	BOEHM ET AL.	(31.01)	789			(0.04)		
				PRODUCTIVITY						[0.05]		
		PROTOTYPE	MAINFRAME	PASCAL	BOEHM ET AL.	(31.01)	232			(0.13)		
				PRODUCTIVITY						[0.07]		
	EXPERT	LINEAR	MAINFRAME	PASCAL	BOEHM ET AL.	(31.01)	498			(0.06)		
		PROTOTYPE	MAINFRAME	PASCAL	BOEHM ET AL.	(31.01)	323			(0.10)		
				PASCAL	BOEHM ET AL.	(31.01)	422			(0.07)		
				PRODUCTIVITY						[0.08]		
				PRODUCTIVITY						[0.08]		
	?	LINEAR	?	COBOL	ALBRECHT	180				0.05		
				COBOL	ALBRECHT	190				0.07		
				COBOL	ALBRECHT	200				0.03		
				COBOL	ALBRECHT	250				0.06		
				COBOL	ALBRECHT	290				0.03		
	?	?	MAINFRAME	COBOL	RUDOLPH	105	990			0.11		
				COBOL	RUDOLPH	145	2805			0.05		
				LINC	RUDOLPH	110	75					1.47
				LINC	RUDOLPH	117	62					1.89
				LINC	RUDOLPH	149	148					1.01
				PRODUCTIVITY						0.05		1.45
			MINICOMPUTER	USER-II	RUDOLPH	187		500			0.37	
				PRODUCTIVITY						0.05	0.37	1.45

Appendix table A4. Development productivity - large applications (300 - 4800 FPA)

SIZE CATEGORY	EXPERIENCE IN LANGUAGE	DEVELOPMENT STRATEGY	HARDWARE ENVIRONMENT	LANGUAGE	RESEARCH	SIZE FPA	WORKING HOURS 3GL	WORKING HOURS 3½GL	WORKING HOURS 4GL	PRODUCTIVITY FPA/hour 3GL	PRODUCTIVITY FPA/hour 3½GL	PRODUCTIVITY FPA/hour 4GL
LARGE	NOVICE	PROTOTYPE	MAINFRAME	ALL	VERNER & TATE	863			9856			0.09
	?	LINEAR	?	COBOL	ALBRECHT	350				0.04		
				COBOL	ALBRECHT	380				0.13		
				COBOL	ALBRECHT	410				0.06		
				COBOL	ALBRECHT	450				0.01		
				COBOL	ALBRECHT	480				0.05		
				COBOL	ALBRECHT	485				0.04		
				COBOL	ALBRECHT	520				0.03		
				PL/1	ALBRECHT	600				0.03		
				COBOL	ALBRECHT	620				0.05		
				PL/1	ALBRECHT	660				0.06		
				PL/1	ALBRECHT	750				0.04		
				COBOL	ALBRECHT	780				0.04		
				COBOL	ALBRECHT	1180				0.03		
				COBOL	ALBRECHT	1900				0.02		
	?	?	MAINFRAME	COBOL	RUDOLPH	436	330					1.32
				COBOL	RUDOLPH	549	660					0.83
				LINC	RUDOLPH	601			495			1.21
				LINC	RUDOLPH	652			900			0.72
				LINC	RUDOLPH	675			660			1.02
				LINC	RUDOLPH	700			660			1.06
				LINC	RUDOLPH	1848			1650			1.12
				LINC	RUDOLPH	1986			2535			0.78
			MINICOMPUTER	COBOL	RUDOLPH	406					0.37	
				RPG III	RUDOLPH	594					0.59	
				USER-II	RUDOLPH	610					0.56	
				RPG III	RUDOLPH	772					0.54	
				COBOL	RUDOLPH	855					0.38	
				COBOL	RUDOLPH	4770					0.28	
				PRODUCTIVITY						0.04	0.45	0.91

Andreas Munk-Madsen, Lars Erik Janlert, Birte Lund,
Eleanor Wynn and others in intensive discussions

PROTOTYPING-BASED EXPERIMENTS IN USER-ORIENTED SYSTEM DEVELOPMENT

Fanny-Michaela Reisin and Daniela Wegge
Technische Universität Berlin
Fachbereich Informatik, Sekr. FR 5-6
Franklinstr. 28/29
D-1000 Berlin 10

0. Introduction

Software development processes are creative work processes. Creative work processes are characterized as seeking and learning processes in the course of which new insights are gained, new ideas developed and, in some cases (as in software development), implemented.

Seeking and learning processes of this sort are empirically supported and promoted by experimentation. This applies generally, not only to software development processes. Experiments serve to test the robustness of ideas and theories on living, tangible reality.

In the case of software development processes that are carried out by users and developers in direct cooperation, computer-supported experiments allowing ideas to be illustrated by computer implementations are very important. Particularly if an open development strategy is pursued they not only allow sensory experience in the – essentially cognitive and language-oriented – work processes, but also have a coordinating and synchronizing function.

An important prerequisite here is that the experimental activities are meaningfully integrated into the work process as a whole. Experiments are not ends in themselves. A conscious decision must be taken as to whether a particular experiment is possible and necessary in a given situation. Other points requiring clarification are the purpose the experiment is to serve, and the way it is to be prepared, conducted and evaluated. A knowledge of systematic prototyping [Floyd 84] and an instructive scheme for practical execution together provide a sound basis for supporting user-oriented software development.

The ideas presented in this paper were developed in the context of our R & D project PEtS (Participative Software Development for Transparency in Computer-Supported Work) [Mehl, Reisin, Schmidt, Wolf 88; PEtS 89] and further elaborated by Daniela Wegge in her intermediate thesis [Wegge 88].

In **Section 1**, we give a brief outline of our daily life in the PEtS project. The aim here is to convey some impression of the living and working milieu, of the people who have spent most of their time over the last two years in this milieu, and of the tasks facing them.

In **Section 2**, we go on to show how we came to the conclusion that prototyping should be used in our project for experimental purposes only. In order to do so, we outline some essential aspects of our methodological approach STEPS (**S**oftware **T**echnology for **E**volutionary **P**articipative **S**ystem Development) which has been tested and further developed in the course of our project [Reisin, Schmidt 88; Floyd, Reisin, Schmidt 89].

We then go briefly into the role we see experiments as playing in such projects. This helps in understanding why the use of prototyping for experimental purposes requires careful consideration, justification and preparation.

In **Section 3**, we take a look at the different types of experiment developed in the context of PEtS and report on examples of the use of these different types.

We were led to the conclusion that experiments based on prototyping are a valuable support in cooperative work between users and developers since:

- we have conducted experiments frequently,
- we have learnt a great deal in the process,
- we have made practical progress, and
- none of us is dissatisfied.

1. Life in PEtS[1]

Geographically speaking, PEtS extends from Berlin to Düsseldorf. Düsseldorf is bordered by the Federal Republic of Germany. It is on the Rhine, and is not far from France, Holland and

1 PEtS: Participative Software Development for Transparency in Computer-Supported Work. This project is funded by the North Rhine-Westphalian Ministry of Labour, Health and Social Affairs under their programme "Mensch und Technik: Sozialverträgliche Technikgestaltung" (Man and Technology: Socially-Oriented Technology Design).

The project is concerned with the development of a software system to support staff of the Central Records Office for Collective Wage Agreements at the German Trade Union Federation's (DGB) Institute for Economics and Social and Sciences (WSI). The staff's daily work consists in registering, filing away and evaluating collective wage agreements.

Both the staff of the Central Records Office and the members of the project team at the TU Berlin are interested in utilizing the advantages of cooperative system design (particularly testing and revision of software versions) for socially-oriented technology development and user-oriented work design.

Belgium. Berlin is bordered by the German Democratic Republic. It is on the Spree, and is not far from Scandinavia.

Historically speaking, the idea for PEtS was born in December, 1986. As a sovereign, autonomous project, funded by the Government of North Rhine-Westphalia, PEtS has been in existence since June, 1987.

PEtS has a total of fourteen permanent residents, nine of them female and five male. There are also a number of non-residents who occasionally visit PEtS.

Of the PEtS residents, seven are "occupied" as users (two of them also heads of department), and seven as developers.

Life in PEtS revolves around four different places:

- The Central Records Office for Collective Wage Agreements at the German Trade Union Federation's (DGB) Institute for Economics and Social Sciences (WSI) in Düsseldorf.
- The research group on Software Engineering at the Technical University of Berlin,
- The large and small conference rooms at the WSI.
- Planes travelling between Berlin and Düsseldorf.

The users spend 40 hours a week of their life at the Central Records Office, the developers spending approximately the same amount of time at the Technical University. Part of the developers' time is spent on planes, which take about an hour from Berlin to Düsseldorf. The developers fly regularly to see the users. The users never visit the developers – a state of affairs which has simply developed in the course of time. The reasons for this are of no further interest here. When the developers are on one of their visits to the users, during which they work, discuss, learn, laugh and argue with each other, this all takes place in either the large conference room at the WSI, or the small conference room at the Central Records Office. It should, however, be added that this small conference room no longer exists as such. About a year ago, it was, along with another room, turned into a computer room. Since then, the developers and the users have spent most of their time working together here.

The Users' Tasks

The Central Records Office for Collective Wage Agreements is a service organized by the DGB. The users register all collective wage agreements concluded (3,600 annually), file them away in the archives (containing a total of 40,000 records), and evaluate them according to specific quantitative and qualitative criteria, e.g. working hours, wages and salaries, rationalization safeguards, professional training provisions, etc. All enquiries received daily by telephone or in writing are answered directly. In addition, monthly, quarterly and annual reports are published in the form of bulletins, assessments, statistics, etc. The services provided by the Central Records Office are used by the general public and by the trade unions themselves, and they are highly thought of even by the employers' associations. The women working in the Records Office are aware of the significance of their work, are themselves convinced of its importance, and feel very much at ease in their working environment.

Before we met them, there was not a single computer in use in the Records Office. Everything was done by hand. Having worked at their jobs for a considerable period (6 - 15 years) and having jointly developed all the work aids in use, e.g. filing systems, evaluation grids, report forms etc., the women are highly proficient at finding specific files and at assessing, evaluating and processing data.

The idea of installing a computer would never have occurred to them if there had not been a marked increase in the number of daily enquiries received, and in the qualitative demands made with respect to evaluation, as a result of the growing intricacy of collective wage bargaining in the Federal Republic. The additional fact that collective wage agreements are becoming more and more specialized, and therefore constantly increasing in number, has also made their work more stressful. They no longer have time for qualitative evaluation or for developing new forms of reports. The Records Office is, nonetheless, compelled to retain a degree of flexibility enabling it to adapt its services to new developments. The future of the Office depends on this.

In its organizational structure, the Records Office is similar to a public authority. Each member of staff has her own office and is responsible for a particular sphere of duties, i.e. for one or more trade unions. The head of department requests data for reports, which she prepares herself, or formalized reports processed accordingly. The heads of department bear sole responsibility for the preparation of reports and in questions of their organization and content, i.e. decision-making, organizational problems, requesting information, etc. The other members of staff support them in these tasks.

The Developers' Tasks

The research group on Software Engineering is engaged in teaching, research and development activities at the Technical University of Berlin. Five of the developers are concerned exclusively with PEtS-related tasks, in particular with the configuration of the development computers and with the construction and implementation of software. Two of the developers are engaged in teaching and research work within the Software Engineering group. The developers working directly on the project share a large office, the other two a smaller one. All decision-making, development of ideas and organizational measures is done jointly by all seven of the developers.

The Developers' and Users' Joint Tasks

The tasks performed jointly by the developers and users comprise the development of computer support for the following user activities:

- Registering and making alterations to collective wage agreements and their relevant criteria.
- Searching for specific wage agreements.
- Evaluating wage agreements according to particular criteria.
- Preparing statistical charts in accordance with specific criteria.

The computer system is to be designed for use in daily work routines. Users should be able to perform their daily work tasks without the need for outside help. It was their express wish to be able to concentrate, after installation of the computer, on the substance and purpose of the activities entailed by their work, without the need to concern themselves with data processing problems.

The basic principle followed in PEtS was that the users were to determine themselves the quality criteria relevant to their work and, in doing so, were not to allow themselves to be influenced by either technical or scientific constraints.

So much for life in the PEtS project.

2. Prototyping-Based Experiments in PEtS

One of the key factors in the feelings of suspicion shown by the users towards us, the developers, was a sense of the existence of a "responsibility gap". As the users saw it, the project and its results were going to create a reality in which they, the users, would have to live and work over a long period. In contrast, we, the developers, would be able to "quit the scene without any serious implications" once the project was concluded. A major part of initial planning discussions revolved around the issue of the developers' credibility in questions of responsibility and personal commitment.

It was against this background that the following situation arose during project establishment: every time we mentioned the word "prototype", the users took it to mean "provisional solution", and their reaction was one of alarm. This resulted in lengthy verbal and written warnings, in assurances, and in explanations being given on both sides. But the scepsis remained. It was this state of affairs that convinced us of the urgent need for a clarification of terms. Here, it was important to distinguish from the start between two different sorts of software products: between those which were to be installed at the users' workplace and which were to serve them as tools for performing their regular work tasks, and those which they were merely meant to try out, i.e. to use on a provisional, experimental basis to help primarily in solving the problems posed by the joint design process.

2.1 Clarification of Terms

Software Version and Software Prototype

The software development process, in which both users and developers participate on a cooperative basis, is concerned with devising and creating a new work area, constituted by a software product and the changed processes and structures in the application area. Since software development, as we see it, is an evolutionary process extending over several cycles of construction and use, we speak of software (product) versions when we mean products that are installed in the computer-supported application area.

The key consideration in the use of software versions is the purpose which the users' daily work tasks serve. Evaluation of the quality of computer support is not the primary objective in using a software version, but rather its indirect result. It is a consequence of changed requirements on the part of the users and/or the user organization during regular use.

Unlike software versions, software prototypes serve experimental purposes only. Prototypes are constructed and used to visualize ideas developed jointly in the course of designing the new work area and implemented on the computer on an experimental basis.

Experimental Prototyping

The processes involved in the production and testing of prototypes are termed by us "experimental prototyping". The epithet "experimental" is used to differentiate our view from other views of prototyping (e.g. "rapid prototyping", "evolutionary prototyping" etc.).

Experimental prototyping is a subactivity in the cooperative software development process, serving to support communicative searching and learning processes. Prototyping, then, supports work processes which result in a software version. This does not mean that prototypes cannot be incorporated as components into the software version. However, in the context of cooperative work processes between developers and users, experimental prototyping should not focus on this – basically technical – aspect, but on gaining insights and experience. This is why the planning, preparation and performance of an experiment is so important.

What do we mean, though, by an experiment, and what sort of experiments are involved in the context of cooperation between developers and users?

Experimentation may be defined as the endeavour to test the realization of an idea or a theory not merely by observation, but by actively changing a given situation. We are talking here about changes in situations where something which does not yet exist is to be "constructed for the first time", or where "elements are selected to form a new constellation" from something which already exists. The ambiguous usage of the phrase "realization of an idea" may stand without further qualification. In actual fact, experimentation is, basically, the same thing as realization of an idea being illustrated in the sense of the construction activity, the constructed results, and the perception activity.

Experimental prototyping means the computer-supported "constructive" or "selective" realization of an idea. Here, we take "constructive realization" to mean the first implementation and testing of an idea, and "selective realization" the tailoring of already implemented components to a specific aspect.

Experimentation is a purposive activity. The experiments must be consciously planned and prepared with a view to their suitability for fulfilling the respective purpose, in order that performance and evaluation of the experiments may, in turn, fulfil their purpose.

Considering experiments as purposive actions, we distinguish them from mere observation and such actions in which an idea is "just given a quick tryout". These actions are part of every creative work process, must always be possible, and are therefore not included in our methodical considerations.

2.2 Experimental Prototyping in the Context of Cooperative Design Processes

It is essential, particularly in the case of cooperative work processes and with a special view to communication, learning and design processes between users and developers, that the purpose of such activities and their planning and execution be given careful consideration. This involves taking into account a number of different points:

Prototyping requires a considerable amount of time and effort on the part of the developers and users.

Good software development tools and environments can do nothing to alter this fact, any more than can a good knowledge of programming.

Experience gained in the course of the PEtS project has shown that careful attention must be paid to the time required for the construction of a prototype ("constructive realization"), however small, e.g. a functional model of a wage agreement in skeleton form (without entries), the functionality of the function "search for wage agreement" or the visualization of several representational variants of "list of wage agreements found".

Even if the prototype in question was a component of the software version which was to be illustrated from certain perspectives ("selective realization"), it took a considerable amount of time to tailor it to the needs of the experiment.

By the same token, the users, too, invest a certain amount of time in preparing and participating in the experimentation sessions. Particularly in cases where the participation of as many users as possible is required, and where the developers' and users' workplaces are physically separated, joint prototyping sessions take several days.

Prototypes are incomplete models of the software version.

Nevertheless, all aspects of the software product are invariably visible, i.e. the screen display, the functionality, the use procedures, the interaction of the operating devices, etc. It is therefore imperative that attention be focussed during prototyping on the relevant aspects of interest, shutting out all others. Otherwise, prototyping has a rather disorienting effect. The users complain about faults or develop ideas that are irrelevant for the developers whose attention is focussed on other problems. Despite their "common" sensory experience, the developers and users talk at cross-purposes.

An equally vague situation is that where developers are open to "all" points of view, i.e. record and evaluate "everything". This leads to fortuitous factors, arising from the respective situation, being incorporated into the evaluation and being taken up again in the subsequent development process. No one is afterwards aware of how such factors came to be taken into consideration.

Furthermore, it must be assumed that the users' perspectivity is rooted primarily in their daily work processes, in other words that they invariably relate a prototype holistically to their own work. Where prototyping lacks a conceptual basis, this may impair the creativity and productivity of the common work processes. The more assurances the developers give about what will be different when the system is later in use, the more disconcerted the users get. Such assurances are intangible compared with the visible, concrete prototype.

Prototyping must be determined by the purpose the prototype is to serve, not by the prototype itself.

The experimental situation in which the users test the prototype is not transparent to them even in cases where the prototype "looks perfect" and "doesn't break down". If the purpose served by, and the procedures involved in, prototyping are not transparent to the users, the results will be largely determined by the design decisions implemented in the prototype. Focussing attention on what already exists restricts imagination and creativity, especially where one perfect solution is demonstrated without any alternatives [Bødker et al. 88]. In actual fact, it is wrong to speak of cooperation in such cases, since the users merely give their approval to the developers' decisions.

Prototyping is social interaction between developers and users.

A common experimental situation involving both users and developers does not come about of its own accord. If the tasks to be assumed by the developers and users during prototyping are not prepared beforehand, there is invariably a danger of the developers' "running" the prototyping session. Owing to the developers' position of superiority as regards performance of the experiments, the users are unable to share in the "common experience". They fail to take advantage of the opportunity to enhance their competence (which would, in turn, enable them to learn to assess for themselves the scope available in designing computer-based systems), and confine themselves instead to asking questions and relying on the answers and demonstrations given by the developers. The users are unable to pursue their goals self-reliantly, and if this was the object of the experiment, it has failed.

From this, it follows that the decision for or against experimental prototyping must be taken with reference to a common, communicative situation and determination of purpose **prior to** preparation and execution of the experiment. The objective must be reflected in the planning and preparation activities in such a way that the results can be interpreted as an answer to the initial questions posed. Here, numerous interference factors have to be identified and controlled.

To summarize then, we are able to pinpoint a number of essential factors which have to be taken into account when making preparations for prototyping:

- The decision to carry out experiments must be geared to the requirements of a given project situation. The purpose of prototyping must be determined in relation to the project situation, and the question of the suitability of prototyping in this context must be looked into.
- The features the prototype is to possess with a view to the objective of the experiment must be determined.
- The prototyping session must be well planned and prepared. This means that the users' and developers' respective tasks must be fixed in advance, and that explicit consideration must be given to both favourable and unfavourable situational constraints.
- The evaluation and incorporation of the results into the overall development process must be worked out in advance.

Experiments and their results are largely dependent on situational constraints and the contextual conditions in which they are carried out. In the context of cooperative software development it is not only material, spatial and temporal conditions that are of significance. In particular, the users' and developers' subjective and collective interests, their social situation etc. must be taken into account when preparing the experiments.

In various areas of experimental research quite considerable attention is paid to the control and manipulation of influential factors. Admittedly, the conditions in experiments from which human beings are largely excluded (merely assuming the role of observers), e.g. in physics, are quite different. For this reason, experiments conducted in the social sciences appeared to us a natural choice in our search for suitable analogous methods. The approaches adopted in social psychology seemed to come closest to our own requirements. Here, consideration is given in designing experiments not only to material and temporal factors, but also to the motives of the persons involved in the experiment and the roles assumed by them. A critical appraisal of the experiments conducted in social psychology[1] can, however, only be regarded as a first step here. It would be equally important to look at experiments in, say, the field of education.

The Basic Experimental Constellation

Our examination of experiments in the field of social psychology enabled us to identify four factors of influence that go to make up the basic constellation of experiments on the support of cooperative learning and communication processes[2]: the degree of strangeness of the object under investigation; the strictness of the instructions given; the test person's freedom of reaction; and the power differential between developers and users. These four factors describe conditions which are critical to cooperation.

The *degree of strangeness of the object under investigation* signifies the subjective distance felt by the users to the experimental environment and to the task assigned them during the experiment. If the aim is for the users to behave without constraint and to participate on equal terms with the developers in the experiment, the attempt should be made to design the experiment in such a way as to minimize its alienating effect. The more unfamiliar the situation, the object under investigation and the users' task is, the more the users put themselves "at the mercy" of the developers in the experiment. They forfeit a portion of their self-responsibility for the duration of the experiment, as they are unable to grasp the situation as a whole. They have to rely on the developers' retaining control of the situation.

A example of a familiar situation is the the users' customary working environment, or the users' having questions put to them in the course of a normal, informal conversation. In contrast,

1 Details are given in [Wegge 88].

2 These four factors or dimensions were first distinguished by Heinz Schuler. He describes in comparable terms the basic social constellation in experiments in psychology [Schuler 80].

confronting a user unversed in data processing with a prototype manifesting features of an operating system might be considered an instance of an unfamiliar situation.

Strictness of the instructions denotes the extent to which the developers provide the users with directions for specific actions. These instructions include all the material provided and stipulations made by the developers with respect to the way in which the experiment is to be conducted. Strict instructions tend to limit the users' and developers' imagination. The frequently cited advantage of strict instructions, i.e. in clearly defining what is to be investigated and allowing the experiment to be precisely tailored to the particular question requiring clarification, does not hold for cooperation between users and developers.

An example of a situation where few instructions are given is one in which the users are simply called on to describe the activities of a particular work area. In contrast, a situation with a high level of instructions might be one in which users are confronted with a complex, fully elaborated model.

Freedom of reaction refers to the degree to which the developers restrict and control the users' behaviour during the experiment. Such restrictions can be of advantage where a specific solution is to be evaluated, and responsibility for the evaluation of the prototyping session lies with the developers. On the other hand, dispensing with control of the users' reactions means that they have a greater chance of gaining personal experience, e.g. in learning to use a prototype, than would have been possible if they had been closely supervised by the developers. A further advantage obtained by restricting control is that the users are able to carry out evaluation of a prototype on a largely autonomous basis.

High freedom of reaction is enjoyed in an experiment where the users are able to test and evaluate a prototype in their own working environment, without supervision or time constraints. Control of reaction is very high in an experiment where the users are required to enter predefined data on a form containing predefined fields in the presence of the developers.

The *power differential* is a factor existing prior to and outside the context of the experiment, it being determined by age and by social and professional status. If the developers are in a powerful position with respect to the users, they may be expected to exert a high degree of influence on the users' behaviour, or conversely, the users may be expected to behave less "normally" towards the developers. We have a situation with a low power differential where developers construct software for developers.

The power differential between developers and users cannot be influenced in the context of the experiment; it exists independently of it. To this extent, it is a factor that is not amenable to planning. The developers should, though, be aware of the fact that they are able to exercise considerably more influence than is perhaps desirable in the context of the particular question requiring clarification. In contrast, the degree of strangeness of the object under investigation is partly dependent on the way the experiment is prepared: prior to the experiment, it is possible to give the users some sort of introduction to the object of the experiment or familiarize them with it. The strictness of the instructions and the users' freedom of reaction are parameters which are totally dependent on the experiment itself. With respect to them, it is essential that the decisions determining the experimental constellation are sound and well-founded.

3. Four Types of Experimental Prototyping

In the course of the PEtS project, our work processes with the users were designed mainly with the help of experimental prototyping. Only in cases involving fundamental strategic decisions were long discussions held and purely written specification documents produced.

Since it was also our intention while working on the project to learn "something about methods", we tried, from the start, to define the type of experiments to be adopted. We did so on the basis of a slightly modified version of the classification first made by Floyd in [Floyd 84]. In retrospect, it was possible to distinguish four different types of experiment that proved useful:

- Exploratory experiments
- Decision-making experiments
- Evaluative experiments
- Pilot experiments.

Exploratory Experiments

Exploratory experiments are used by developers and users in their joint endeavour to clarify conceptual problems. Neither the developers nor the users have a clear idea of what the future computer-based system is to do. The developers' knowledge of the application area is too limited, and the users have difficulty in assessing the sort of jobs a computer-based system is capable of performing. As far as the experiments themselves are concerned, this means that, in the preparatory stages, it would be necessary to consider a vast number of alternatives. It also means that particular attention should be paid to conditions providing for open discussion of potential solutions between users and developers.

In the case of exploratory experiments, it should be borne in mind that the users are restricted in their scope of action both by the degree of strangeness of the object under investigation and by the complexity of the instructions given by the developers. For the users, every prototype constitutes a complex model with which they must first of all come to terms. This makes it difficult for the users to incorporate their own ideas and experience at all. Especially in the initial phase of development, there is a danger of the users' ideas being too strongly shaped and thus limited by a prototype.

Where a situation is relatively open and the aim is to give concrete illustration to a vague idea in order to obtain as much new information as possible, it is a good idea to narrow down the experiment's objective as much as possible. This means that, although "free rein is given to the imagination", the domain of discourse relating to the developers' and users' communication and learning processes remains limited.

In PEtS, exploratory experiments have generally been used in the early stages of constructive design. At the outset of the project, for instance, we used exploratory experiments to obtain a suitable model of the wage agreement. We did so, together with the users, by alternately describing the structure of the agreement, and then visualizing it on the computer with the help of a data model. Using selected data entries of a usual and unusual nature, the users were able to test whether all aspects of a wage agreement had been covered, and at the same time familiarize themselves with the new structure. The developers, for their part, were able to test the soundness of the data model.

Decision-making Experiments

When designing computer-related work routines, there are frequently several alternatives under discussion with respect to the configuration of operating devices, visualization of data and functions on the screen, or the structuring of the functional sequences. Various design options are available, for instance, where the users formulate different requirements as a result of differences in working styles or interests, or where the developers devise several potential solutions in the course of their communication processes.

Here, it is possible to restrict the number of alternatives which may be selected. The prototypes used should be evaluated with specific reference to the aspect on which a decision is required. The discussion between users and developers is therefore no longer an open one, but is, instead, geared to the design of a particular aspect. If an experiment is intended to help determine what all the users consider the soundest alternative, it is essential to specify precisely which of the options are suitable, and where the differences between them lie.

Where a number of potential alternatives may be realized in the form of prototypes, the result of the selection process is a decision in favour of one of these alternatives. In the decision-making experiment, the users are required to make an explicit choice between different models. This does not necessarily mean that only one particular view must be realized. Decision-making experiments are designed to help identify what may be considered sound alternatives from the users' point of view. They may result in either one single option and view being realized, or several different ones. Prior to the experiment, it is important to settle the question of whether the users and developers wish to confine themselves to one single option, or whether several different ones are to be realized.

An instance of the use of a decision-making experiment in PEtS was for allocating functions to the various input devices. The important thing here was that the various options should not only reflect the different possible ways of distributing operator functions between mouse and keyboard, but that they should also illustrate the interaction between the various functions for each different constellation.

Evaluative Experiments

Evaluative experiments are appropriate where a implemented component is to be integrated into the target version. Here, only one single model or prototype is evaluated in the course of the experiment. Restricted experiments of this sort can, ultimately, have only two possible outcomes: the users either make a detailed assessment of the model, suggesting various local modifications; or they reject the model outright (this latter case is an indication of poor design of the development process up to this point).

In the context of PEtS, the object of using evaluative experiments, for instance to test the component "search for wage agreement", was to determine, by reference to the interaction between functionality, visualization and operating devices, whether the computer-related working routines were appropriate to the tasks in hand and capable of meeting the users' quality criteria. The use of evaluative experiments generally made it possible to clear up numerous questions of detail and local inconsistencies.

For both decision-making and evaluative experiments, it is imperative that a set of instructions be given and control facilities be provided for in each case. The aim of the experiment is, after all, to evaluate specific aspects of the prototype. It is therefore important that the purpose and execution of the experiment be transparent for the users, and that interference factors be excluded as far as possible. The users are only likely to be able to perceive problem-related aspects if they do not experience a sense of alienation when operating the prototype and observing its visible behaviour.

The procedural plan developed for prototyping experiments sets out to cover the individual steps and decisions taken during exploratory, decision-making and evaluative experiments. Here, particular attention is paid to the conditions promoting or obstructing cooperation.

Pilot Experiments

Pilot experiments present the user with the opportunity to use and evaluate a particular prototype over a relatively long period, in their accustomed working environment, and subject to their own control. There is little intervention on the part of the developers in a situation of this sort.

Where the implemented component involves processing of an extensive subtask, it is a good idea to arrange for pilot experimentation before the component is integrated in the software version. The essential thing here is to point out to the users the factors which are to be considered in testing the prototype, and to agree on how the developers are to be informed of the results of the test. One way which proved suitable, especially for the pilot experiments we conducted in PEtS, was feedback by telephone. Pilot experiments are an extremely important method here, especially where the software components being tested relate to completely new tasks.

In PEtS, it was not until pilot experiemts were carried out on the software component "updating employment figures" that the users realized the advantages and disadvantages of updating the employment figures for all wage agreements in a single computer-supported work routine. Another factor which became evident to them in the process was the potential rationalization effect here - an aspect which was disregarded during design and which they were now able to make an issue of.

4. Summary

The decision for or against a particular type of experiment depends both on the concrete project situation and on general factors relating to the state of development as a whole and on the technical and material resources and means available.

No attempt is made in the present paper to consider the basic technical requirements for using prototyping. The discussion on suitable tools for constructing prototypes is being conducted on a relatively broad basis compared with the discussion on suitable procedures for prototyping.

The experience gained in PEtS has shown that the decision as to which of the various types of experiment are most appropriate in a given situation is dependent on how far development of a particular product component has progressed, and hence on how perfect our knowledge is of what the design and modelling processes set out to do. Exploratory and decision-making experiments play a crucial role in specifying the activities involved in a particular work area and developing conceptual solutions for them. In contrast, evaluative and pilot experiments are of greater importance for implementation of the product and its transition to the application area.

References

[Bødger et al. 88] S. Bødger, J. Lindskov Knudsen, M. Kyng, P. Ehn, K. Halskov Madsen: *Computer Support for Cooperative Design.* Computer Science Department, Aarhus University, 1988.

[Floyd 84] C. Floyd: *A Systematic Look at Protoyping.* In: R. Budde, K. Kuhlenkamp, L. Mathiassen, H. Züllighoven (Eds.): *Approaches to Prototyping*, Springer Verlag, Berlin, Heidelberg, New York, Tokyo, 1984.

[Floyd, Reisin, Schmidt 89] C. Floyd, F.-M. Reisin, G. Schmidt: *STEPS to Software Development With Users.* To be published in the Proceedings of ESEC 1989, Sept. 11-15, 1989, Warwick, England.

[Mehl, Reisin, Schmidt, Wolf 88] M. Mehl, F.-M. Reisin, G. Schmidt, G. Wolf: *Prototyping als Technik im Kontext partizipativer Systementwicklung.* In W. Brückers, N. Meyer (Eds.): Zukunftsinvestitionen Berufliche Bildung. Vol. 3, Berufsförderungszentrum Essen e. V., Cologne, 1988.

[PEtS 89] C. Floyd, M. Mehl, F.-M. Reisin, G. Schmidt, G. Wolf: *Zwischenbericht des Projekts PEtS.* Technical University of Berlin, 1989.

[Reisin, Schmidt 88] F.-M. Reisin, G. Schmidt: *STEPS - Ein Ansatz zur Evolutionären Systementwicklung.* Computer Magazin, Vol. 17, No. 7/8, 1988.

[Schuler 80] H. Schuler: *Ethische Probleme psychologischer Forschung.* Verlag für Psychologie - Dr. C.J. Hogrefe, Göttingen, Toronto, Zürich, 1980.

[Wegge 88] D. Wegge: *Experimentmodelle und methodische Konzepte zum Einsatz von Prototyping im Kontext partizipativer Systementwicklung.* Technical University of Berlin, 1988.

THE MISSING CONCEPTS OF USER PARTICIPATION: An empirical assessment of user participation and information system success

Timo Saarinen
and
Markku Sääksjärvi

Helsinki School of Economics
Runeberginkatu 14 - 16
SF-00100 Helsinki
FINLAND

ABSTRACT

Many authors have hypothesized that user participation is one of the most important factors affecting the success of information system development (ISD) projects. However, published empirical studies have given contradictory evidence. In this paper we argue that the diverging results can be explained by conceptual and methodological limitations of these studies, and that user participation cannot be evaluated alone, without a parallel analysis of both users and systems analysts contribution to the success.

In this paper we present a framework to study the effect of user participation on information system success. We use the framework for an empirical assessment of 48 large ISD projects in leading Finnish organizations. On the basis of our analyses we argue that the quality, not the quantity, of participation is of key importance. A good balance of both users and system analysts participation and competence are needed so that all phases of the development life-cycle succeed.

On the basis of the analyses we believe that user participation can be of great value but we strongly recommend that the practitioners emphasize the quality of user participation instead of relying on its magic power.

Key words: Participation, skills, information system success

1. INTRODUCTION

User participation has been recognized to be among the most important factors affecting the success of information system development (ISD) projects. However, empirical studies have found contradictory evidence (Hirschheim 1983, Ives and Olson 1984, Lyytinen and Hirschheim 1987). As Ives and Olson (1984) argued the benefits of user participation have not been convincingly demonstrated in academic research. They concluded that the majority of the studies have been methodologically flawed and the researches have not been able to show the hypothesized relationship between participation and success.

It is not difficult to find shortcomings in the empirical tests of user participation. On the one hand, the frameworks behind the measurements of the key concepts of participation or success have often been inadequate. On the other hand, a search for the possible causal relationship on the basis of aggregated variables may have lead to contradictory results.

The aim of this paper is to find out whether the hypothesized positive association between user participation and ISD projects' success really exist or not. In order to do this we propose a more comprehensive instrument to measure the impact of participation on ISD project success than used in earlier studies. In our instrument we take into account not only the users but also the systems analysts and their ability to cooperate and communicate with users. We also enlarge the measurement of the success of an information system from the well known User Information Satisfaction (UIS) measure to a three component instrument which also includes the development process and organizational impact factors. We test our framework using empirical data collected from 48 large development projects.

Our results indicate that the positive effect of user participation on the ISD project success depends much more on the qualitative than quantitative factors of participation. The factors describing the quantitative aspects of user participation had no significant correlations with the success measures while users' commitment and skills correlated significantly with many of the most important success variables. But it seems to be even more important that system analysts' participation and their communication skills have to be taken into account when analyzing the relation between participation and success of an information system.

2. EARLIER RESEARCH

2.1. ISD project success

There are many different views of what the concept ISD project success really means and how it should be measured (Cherveny and Clark 1981). There is no simple measure for success. The problems arise because even simple cost benefit analyses of information systems are often difficult or even impossible to perform (Ives and Olson 1984) and thus researchers have been forced to develop surrogate measures for the success. They are believed to give the same result as the return on investment (ROI) would give.

One of the most promising result of this research is the development of a standard measurement instrument called user information satisfaction (UIS - Pearson 1977, Bailey and Pearson 1983). Bailey and Pearson ended up with a 39 item instrument out from more than 600 candidates. They developed four scales or adjective pairs for each of the scales. Ives et al (1983) refined this instrument. They removed the unnecessary items and scales from the instrument and ended up with 13 items and only two scales for each. This short form UIS instrument has been used in many IS studies as a dependent variable (for example Mahmood and Becker 1985, Miller and Doyle 1987, Koh and Lawrence 1988). UIS has also come in for criticism (Chismar et al 1985, Treacy 1985 and Iivari 1987). It has been criticized because of its theoretical weaknesses and because it has not been able to pass all the tests in the studies replicating the original study. Treacy (1985) found it unreliable but Baroudi and Orlikowski (1986) reliable. On the one hand, UIS can also be regarded as quite a limited instrument lacking many of the management oriented issues; on the other hand, it is quite a fuzzy instrument including both detailed information attributes and general statements about participation, service level and relationships between user and DP departments. There have also been attempts to develop new UIS instruments (Doll and Torkzadeh 1988, Similä 1988), but they are still in their early stages. In spite of the criticism we believe UIS is one of the best surrogate measures for the quality of an information system.

Besides UIS many other success measures have been used in implementation research; usage of system, perceived success, impacts, changes, level of adaptation, system effectiveness, realized goals, payoffs and performance for example (Ein-Dor and Segev 1978, Hamilton and Chervany 1981, Lucas 1981, Ein-Dor et al 1984, Sanders 1984, Barki and Huff 1985, Kivijärvi 1987, Lyytinen and Hircshheim 1987). Budget and time overruns are often used as process related success measures (Lucas 1981, McFarlan and McKenney 1983). Ives and Olson (1984) also found system acceptance as a common success variable. However, today, at least in Scandinavia, this measure seems to be out of date. Resistance to adapt new technology is not high anymore. Lyytinen and Hircshheim (1987) synthesized much of earlier research by developing a model for IS success. They used a concept expectation failure (usually taken as an opposite for success), which they argue covers all relevant success measures including the correspondence failures, process failures and interaction failures.

Although the measurement of the success of information systems has been an object of active research, it is still quite problematic. The measurement of success cannot be based on any single variable but it has to be based on a multi-dimensional approach taking various aspects of success into consideration.

2.2. User participation

User participation can be defined as "the participation in the system development activities by a member or members of the target user groups" (Olson and Ives 1981 p. 184). Mumford (1979) classifies user participation according to its depth; consultative, representative and consensus type participation. The users should have real power, not only symbolic participation in the development process and this is best achieved by a consensus type participation. Also Edström (1977), Ginzberg (1978) and Ives and Olson (1984) have emphasized that only when users have real influence on the development project user participation has positive effect on the success. In some studies user participation has also been seen as workers' democratic right to influence their own work (Björn-Andersen and Hedberg 1977, Mumford 1979). This is expected to increase the acceptance of the system as well as its quality. On the other hand, Dickson and Simmons (1970) and Guthrie (1972) have recognized management participation as essential for success.

According to Ives and Olson (1984) the belief that user participation leads to increased success can be traced to theories of organizational behavior, especially to the theories of participative decision making and planned organizational change. They found that the results from empirical studies are, however, contradictory.

We argue that one reason for the contradictory results is the vague and narrow conceptualization of both user participation and success. In many studies user participation has been measured by self-rating questionnaires (see a review in Ives and Olson 1984). Questions have usually been posed in Likert-type scales. The emphasis has often been on the evaluation of the average level of user participation in general in the organization or by the respondent. There are also some studies (Henderson 1988, Baroudi et al 1986) which have made attempts to take a step forward in measuring user participation by the mechanisms enabling participation in each of the phases of the development life-cycle. Besides these methodological problems, it may be that the benefits of user participation are dependent on the characteristics of each development project. Certain types of systems and development situations require heavy user participation while others do not.

We believe that the situational adequacy of participation is of great importance. However, there is one more important factor to be considered. As De Brabander and Edström (1977) have emphasized the importance of system analysts' competence for a successful ISD project. This indicates that user participation cannot work if the system analysts are not skilful and cannot communicate and cooperate with the users. To get better understanding of the participation, thus, also the system analysts' participation should be taken into account.

Barki and Hartwick (1989) made a distinction between the concepts user participation and user involvement. This is because most of the research studying the effects of user involvement has actually operationalized the concepts user participation and user involvement in the same way. However, user involvement is merely "a subjective psychological state reflecting the importance and personal relevance of a system to the user" (Barki and Hartwick 1989, p.53) and therefore quite different from what is usually meant by user participation. Empirical studies have not been able to show whether the effects are due to participation or involvement and commitment and thus the results have remained, so far, somewhat fuzzy.

3. RESEARCH FRAMEWORK

On the basis of the literature review we developed a research framework for the empirical assessment of user participation. The framework which is illustrated in figure 1 allowed us to study the relationship between both users' and systems analysts' participation and the project success, and also to take into account the skills of both groups.

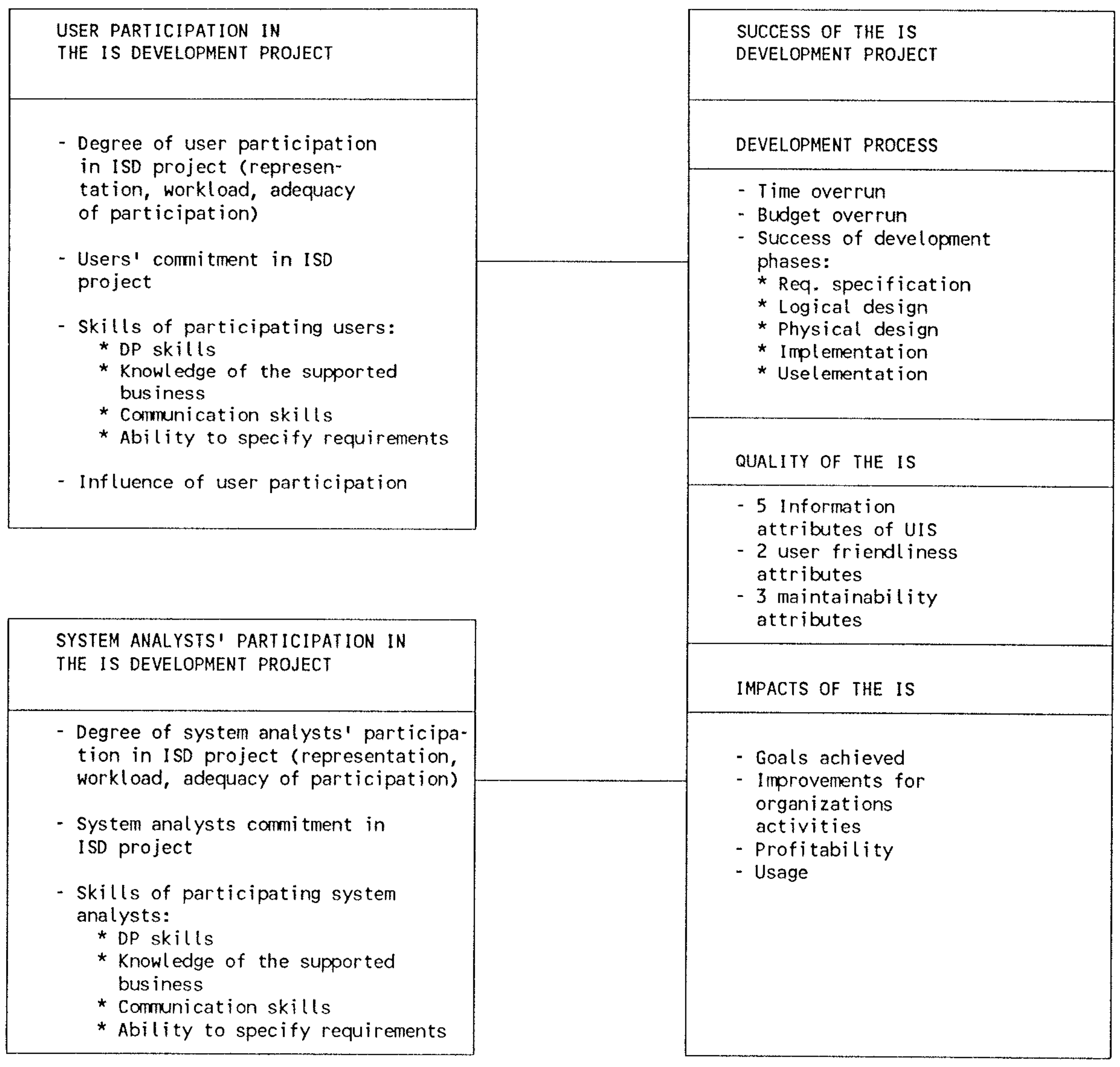

USER PARTICIPATION IN THE IS DEVELOPMENT PROJECT
- Degree of user participation in ISD project (representation, workload, adequacy of participation)
- Users' commitment in ISD project
- Skills of participating users:
* DP skills
* Knowledge of the supported business
* Communication skills
* Ability to specify requirements
- Influence of user participation
SYSTEM ANALYSTS' PARTICIPATION IN THE IS DEVELOPMENT PROJECT
- Degree of system analysts' participation in ISD project (representation, workload, adequacy of participation)
- System analysts commitment in ISD project
- Skills of participating system analysts:
* DP skills
* Knowledge of the supported business
* Communication skills
* Ability to specify requirements
SUCCESS OF THE IS DEVELOPMENT PROJECT
DEVELOPMENT PROCESS
- Time overrun
- Budget overrun
- Success of development phases:
* Req. specification
* Logical design
* Physical design
* Implementation
* Uselementation
QUALITY OF THE IS
- 5 Information attributes of UIS
- 2 user friendliness attributes
- 3 maintainability attributes
IMPACTS OF THE IS
- Goals achieved
- Improvements for organizations activities
- Profitability
- Usage

Figure 1. The framework of the study

3.1 ISD project success

In our framework the ISD project success was measured by a three component instrument consisting of the development process, the quality of the resulting information system, and the impact of the information system on the organization (Saarinen 1988, Saarinen and Sääksjärvi 1989).

In the following paragraphs the most important ISD project success variables of each three group are described. The detailed variables are described in Appendix 1.

Success of the development process

The success of the development process was evaluated on the basis of the budget and time overruns, and on the basis of the perceived success of phases of the development life-cycle.

Quality of the resulting information system

The quality of the resulting information system was evaluated on the basis of the information attributes of the short-form UIS measurement instrument (Ives et al 1983), easiness of the system's use, user friendliness, and maintainability.

Organizational impacts

The impacts of the resulting information system to the organization were evaluated by the achievement of the development goals, usage level of the system, changes (improvements) of the work processes, and the profitability of the system.

3.2 User and systems analyst participation

User participation was measured by both quantity and quality related variables. As quantitative indicators we used the extent of user representation (number of users participating in the project in relation to the total number of users of the final system), users' share of the total workload of the project, and the adequacy of participation. The quality of participation was measured by the users' commitment in the project, skills and knowledge, and ability to communicate and specify requirements.

Systems analysts' participation was measured in the same way as above, both quantitatively and qualitatively.

In order to improve the objectivity of our evaluation we did not base the estimation of the important qualitative variables on a self ranking procedure but we asked the project managers and user managers to evaluate the skills of both system analysts and users participating in the project.

Also the values of the success variables were collected from both users and project amangers. We asked the project managers to evaluate the success variables of the development process, and the user managers to evaluate the two other components of our success instrument, the quality of the system and the organizational impacts.

4. METHODOLOGY

The data of this study was collected from the 200 largest companies and 25 largest banks and insurance companies in Finland (Saarinen and Sääksjärvi 1989). Altogether 272 IS managers from these organizations were contacted by mail and asked to contribute to our study. One short questionnaire was sent to the IS managers where we asked for a list of all projects finnished in the last two years with a brief evaluation of their success. We asked the IS managers to name the two latest implemented information systems for our review, and asked them to name the project managers and the user manager responsible for the system such that we could send another questionnaire to them.

Within one month 102 IS managers returned the questionnaire (around 40 percent response rate). They indicated altogether 247 information systems and selected 101 of these for our evaluation. 70 project managers and 62 users responded to our questionnaire. This response rate was a result of two separate reminder letters and several phone calls. We got 48 responses which contained evaluations from all of the three respondent groups. Detailed tests of these projects assured us that we had a representative sample of all finished 249 projects in large Finnish companies with regard to both the size and type of a system, application area, and overall level of success (Saarinen and Sääksjärvi 1989).

5. RESULTS

5.1. Profile of the participating companies and the information systems

In 1988 the average net sales of the companies studied was 1906 FIM per year (FIM 1 = 0.25 US$), and the average employment 2317. Around two thirds of the companies were from industry and one fifth were retail and wholesale companies. Less than ten percent of the companies came from banking or insurance, and the rest from the service sector.

The average budget of the information systems studied was 1.2 million FIM, the average duration of the development project around 17 months, and the average of the total man months 41. 70 percent of the information systems were transaction systems and 30 percent supported decision making or management control. One third of the systems under study were designed to support accounting and one third marketing. Every fifth system was supporting manufacturing activities, and the rest of the systems business administration or purchasing.

5.2 Impact of participation on the success of the development process

Correlations between users and system analysts' participation variables and the success measures of the development process are given in figure 2.

The quantitative variables of user participation, extent of representation and users share of the total workload, have no significant correlations with the project success variables.
It also seems clear that the quantitave measures of systems analysts' participation are not enough to determine the success of the project. Instead, the qualitative factors, the commitment and adequacy of participation (the measure of how sufficient the participation was), correlate significantly with many of the project success variables. The adequacy of users' participation seems to be a critical factor in project success.

	Time overrun	Budget overrun	Req. specif.	Logical design	Physical design	Imple- mentation	Use
USER PARTICIPATION							
Extent of representation	.07	.23	.23	.10	.05	.10	.05
Users' workload	-.23	-.13	.03	-.14	.15	.18	.01
Adequacy of participation	-.19	-.19**	.47***	.38**	.36**	.36**	.26*
Users' commitment	-.25*	-.23**	.32**	.15	.29**	.40***	.34**
Users' DP skills	-.21	-.15	.22	.09	.00	.06	.03
Users' knowledge of supported business	.14	-.08	.02	-.05	.00	.06	.12
Users' communications skills	-.27*	.05	.23*	.02	.18	-.04	-.03
Users' ability to specify requirements	-.07	.01	.21	-.10	.10	-.10	-.19
Influence of user participation	.15	-.03	.20	.21	.22	.19	.26*
SYSTEM ANALYSTS' PARTICIPATION							
Extent of representation	-.38**	-.06	.06	-.17	.25*	.16	.15
System analysts' workload	-.18	.09	.03	.05	-.02	.09	.16
Adequacy of participation	-.37**	-.13	.22	.26*	.50***	.26*	.25*
System analysts' commitment	.00	-.08	-.13	.10	.19	.24*	.35**
System analysts' DP skills	.00	-.06	-.05	-.05	.23	.22	.34**
System analysts' knowledge of supported business	.00	.00	-.13	.05	.19	.23	.30**
System analysts' communications skills	-.13	-.06	-.28*	-.12	.12	.28*	.39***
System analysts' ability to specify requirements	.01	.09	-.28*	.02	.12	.17	.32**

Figure 2. Correlations between user and system analysts' participation and process success variables (significance level *** = .01, ** = .05, * = .10)

This might indicate that not a lot of users' work but rather a good balance between users' and systems analysts' workloads is needed. In addition to that, the systems analysts must be able to communicate effectively with the users in order to understand their needs and transform them into a succesful information system.

On the basis of our analyses we believe that too many participating users can even cause severe problems to the development project. Similarly, too heavy system analysts' participation in the requirement specification phase may decrease the level of success of this phase.

	Completeness	Relevancy	Precision	Accuracy	Reliability
USER PARTICIPATION					
Extent of representation	-.22	-.19	-.09	.04	-.12
Users' workload	-.03	.11	.24*	.18	.17
Adequacy of participation	-.20	-.01	.15	.27*	.13
Users' commitment	-.26*	.02	.29**	.31**	.21
Users' DP skills	.22	.32**	.11	.00	.19
Users' knowledge of supported business	.06	.19	.11	.12	.20
Users' communications skills	-.06	.00	-.05	-.05	-.01
Users' ability to specify requirements	-.23	-.13	-.04	-.03	-.08
Influence of user participation	.29	.55***	.17	.01	.28*
SYSTEM ANALYSTS' PARTICIPATION					
Extent of representation	-.17	.04	.14	.15	.17
System analysts' workload	-.04	.19	.11	.17	.20
Adequacy of participation	.12	.13	.40***	.33**	.30**
System analysts' commitment	.22	.34**	.24*	.10	.18
System analysts' DP skills	.08	.29**	.16	.09	.08
System analysts' knowledge of supported business	.20	.18	.26*	.17	.20
System analysts' communications skills	.27*	.47***	.29**	.10	.23
System analysts' ability to specify requirements	.17	.27*	.10	.03	.00

Figure 3. Correlations between user and system analysts' participation and information attributes of UIS (significance level *** = .01, ** = .05, * = .10)

5.3 Participation and the quality of the IS

Correlations between participation variables and the information attributes of the UIS instrument are given in figure 3.

Again, the quantitative measures of participation, the extent of participation and share of the total workload seem to have no significant impacts on the quality of the IS. Instead, the adequacy of systems analysts work seem to be a good predictor for the quality of the information content of the system. Users' commitment and system analysts' communication skills are also significantly correlated with many of the quality measures.

It seems that the quantity of users' participation alone will not improve users satisfaction with the system. Instead, the communication skills of the participating systems analysts are of key importance to the quality of an information system.

Correlations between participation and the measures of user friendliness and maintainability of the system are given in figure 4. These success factors seem to be significantly correlated with all system analysts' skills variables. By contrast, only a few of the users' skill variables seem to affect the quality of the resulting information system. Skilful analysts seem to be more important even to the user friendliness of the system than skilful users. It might also be that quite often the projects lack professional people. However, users' knowledge of the supported business, their communication skills, and ability to specify requirements are important for the maintainability of an information system.

5.4. Participation and impacts of the system

Figure 5 illustrates the correlations between the participation variables and the variables measuring the impacts of the information systems on the organization.

Again, the quantitative variables of participation have no significant correlations with the impact variables. On the other hand, adequacy of user participation and users' commitment are significantly correlated with the achievement of the development goals and the positive changes caused by the system (improvements).

	Ease of use	User friend-liness	Error correc-tion	Implem. of changes	Adapt. to new requir.
USER PARTICIPATION					
Extent of representation	-.12	-.23	.08	.03	-.07
Users' workload	-.19	-.21	.18	-.09	.04
Adequacy of participation	.01	.07	.25*	.05	.02
Users' commitment	.02	.05	.29**	.12	.09
Users' DP skills	-.08	.14	.11	.15	.13
Users' knowledge of supported business	.04	.02	.39***	.27*	.35**
Users' communications skills	-.08	.03	.27*	.18	.13
Users' ability to specify requirements	-.15	-.15	.30**	.13	.12
Influence of user participation	.22	.55***	.05	.18	.21
SYSTEM ANALYSTS' PARTICIPATION					
Extent of representation	-.22	-.12	-.01	-.07	-.07
System analysts' workload	.26*	.26*	.20	.21	.03
Adequacy of participation	.12	.30**	.39***	.16	.30**
System analysts' commitment	.36**	.43***	.39***	.28*	.37**
System analysts' DP skills	.36**	.34**	.43***	.30**	.39***
System analysts' knowledge of supported business	.35**	.39***	.34**	.25*	.38**
System analysts' communications skills	.44***	.40***	.47***	.46***	.66***
System analysts' ability to specify requirements	.60***	.38**	.31**	.20	.34**

Figure 4. Correlations between user and system analysts' participation and user friendliness and maintainability attributes (significance level *** = .01, ** = .05, * = .10)

	Goals achieved	Improve-ments	Profi-tability	Usage
USER PARTICIPATION				
Extent of representation	-.23	.17	.03	.15
Users' workload	.00	-.13	.14	-.10
Adequacy of participation	.39***	.50***	.10	.24*
Users' commitment	.35**	.40***	.18	-.06
Users' DP skills	.24*	.09	.05	.00
Users' knowledge of supported business	.13	-.03	.11	.02
Users' communications skills	.19	.12	-.06	-.10
Users' ability to specify requirements	.05	.05	-.13	-.16
Influence of user participation	.37**	.30**	.23	.00
SYSTEM ANALYSTS' PARTICIPATION				
Extent of representation	-.04	.01	.27*	-.05
System analysts' workload	.17	-.04	-.03	-.07
Adequacy of participation	.27*	.27*	.06	.15
System analysts' commitment	.37**	.36**	.45***	-.09
System analysts' DP skills	.41***	.46***	.26*	.16
System analysts' knowledge of supported business	.31**	.45***	.43***	.12
System analysts' communications skills	.51***	.31**	.37**	.15
System analysts' ability to specify requirements	.28*	.35***	.34**	.16

Figure 5. Correlations between user and system analysts' participation and impact variables (significance level *** = .01, ** = .05, * = .10)

System analysts' skills, commitment and adequacy of their participation seem to be of great importance for the success of ISD project, even more important than user participation. There is a significant correlation even between the systems analysts' skills and the profitability of the system. This may, at least partly, come from the fact that the most competent system analysts are allocated to the projects, which are financially most promising.

6. CONCLUSIONS

On the basis of the above analyses it is evident that the influence of user participation on the success of an information system cannot be evaluated on the basis of only quantitative measures. "How much user participation" seems to be an irrelevant question. Instead, the quality of participation, users commitment and their skills, are important for for success.

It was also clear that system analysts' participation, their communication skills and competence must be evaluated in parallel with user participation in order to get a valid picture of the relationship between participation and information system's success.

We are sure that the earlier contradictory empirical evidence can be explained by these two important missing factors. Despite the fact that in our analyses systems analysts participation seemed to dominate users participation we strongly recommend the practitioners to promote user participation but at the same time to emphasize much more the quality of user participation instead of just relying on its magic power.

REFERENCES

Bailey J.E. & Pearson S.W., Development of a tool for measuring and analyzing computer user satisfaction, Management Science, Vol 29, No 5, 1983

Barki H. & Huff S.L., Change, attitude to change, and DSS success, Information & Management, Vol 9, 1985

Barki H. and Hartwick J., Rethinking the Concept User Involvement, MIS Quarterly, March 1989

Baroudi J.J. & Olson M.J. & Ives B., An empirical study of the impact of user involvement on system usage and information satisfaction, Communications of the ACM, March, 1986

Baroudi J.J. & Orlikowski W.J., A Short-form measure of UIS: A psychometric evaluation and notes of use, Journal of Management Information Systems, Vol 4, No 4, Spring, 1988

Björn-Andersen N. and Hedberg B., Designing Systems in an Organizational Perspective, TIMS Studies in the Management Sciences, 5, 1977

Cerveny R. & Clark T., Conversations on "Why information systems fail - and what can be done about it", Systems, Objectives, Solutions, Vol 1, 1981

Chismar W.G. & Kriebel C.H. & Melone N.P., A criticism of information systems research employing "User satisfaction", Graduate School of Industrial Administration, Carnegie-Mellon University, WP 24-85-86, October 1985

De Brabander B. & Edström A., Successful information system development projects, Management Science, Vol 24, 1977

Dickson G.W. and Simmons J.K., The Behavioral Side of MIS, Business Horizons, 13, 4, August 1970

Doll W.J. & Torkzadeh G., The Measurement of end user computing satisfaction, June 1988

Edström A., User Influence and Success of MIS Projects, Human Relations, 30, 1977

Ein-Dor P. & Segev E., Organizational context and the success of management information systems, Management Science, Vol 24, 1978

Ein-Dor P. & Segev E. & Blumenthal D. & Millet I., Perceived importance, investment and success of MIS, or the MIS zoo-an empirical investigation and a taxonomy, Systems, Objectives, Solutions, Vol 4, 1984

Ginzberg M.J., Steps Towards More Effective Implementation of MS and MIS, Interfaces, Vol.8., No. 3, 1978

Guthrie A., A Survey of Canadian Middle Managers' Attitudes Toward Management Information Systems, Charleton University Press, Ottawa, Ontario, December 1972

Hamilton, S. & Chervany N.L., Evaluating Information System Effectiveness: Part I:Comparing Evaluation Approaches, MIS Quarterly, September, 1981

Henderson J.C., Involvement as a Predictor of Performance in I/S Planning and Design, CISR WP No. 175, Sloan School of Management, Massachusetts Institute of Technology, August 1988

Hirschheim R.A., Assessing participative systems design: Some conclusions from an exploratory study, Information & Management, Vol.6, 1983

Iivari J., User information satisfaction (UIS) reconsidered: An information system as the antecedent of UIS, International Conference on Information Systems, December, 1987

Ives B. & Olson M.H., User involvement and MIS success: a review of research, Management Science, Vol 30, May, 1984

Ives B. & Olson M.H. & Baroudi J.J., The Measurement of User Information Satisfaction, Communications of the ACM, Vol 26, No 10, Oct, 1983

King W.R., Strategies for Success in Management Information Systems, Management Decisions, 1979, 417-428

Kivijärvi H., Implementing Model Oriented Decision Support Systems, Dissertation, Series A:53, Helsinki School of Economics, 1987

Koh H.L. & Lawrence M.J., Effects of user involvement and top management support on MIS success; Australian evidence, Proceedings of the ACC'88 Conference Towards 2000, Sydney, 1988

Lucas H.C., The key to Successful Information Systems, Columbia University press, New York, 1981

Lyytinen K. & Hirschheim R., Information Systems Failures - a survey and classification of the empirical literature, Oxford Surveys in Information Technology, Vol 4, Oxford University Press, 1987

Mahmood M.A. and Becker J.A., Impact of Organizational Maturity on User Information Satisfaction with Information Systems, Proceedings of the ACMs 21th Annual Computer Personnel Research Conference co-sponsored with Business Data Processing, May 2-3, 1985, (ed. James C. Wetherbe).

McFarlan F.W. and McKenny J.L., Corporate Information Systems Management - The Issues Facing Senior Management, Irwin, 1983

Miller J. & Doyle B.A., Measuring the Effectiveness of Computer Based Information Systems in the Financial Services Sector, MIS Quarterly, March, 1987

Mumford E., Consensus System Design: An Evaluation of this Approach, in Design of Computer Based Information Systems, N. Syberski and E. Grochla, Sitjhoff and Noordhoff, Gromingen, 1979

Olson M.H. & Ives B., User involvement in system design: An empirical test of alternative approaches, Information & Management 4, 1981

Pearson S., Measurement of Computer User Satisfaction, Ph.D. Dissertation, Arizona State University, 1977

Saarinen T., System Development Methodology and Project Success: An empirical assessment of contingency models, Proceeding of the 11th IRIS Seminar, Röros, Norway, Aug 1988.

Saarinen T. & Sääksjärvi M., Success of Information Systems in Large Finnish Organizations, Helsinki School of Economics, Series D:116, May 1989

Sanders G.L., MIS/DSS success measure, Systems, Objectives, Solutions, No 4, 1984

Similä J., Modelling and analyzing empirically the success of ADP systems use. Acta Universitatis Ouluensis, Series A, Scientiae Rerum Naturalium 196, Oulu, 1988

Sääksjärvi M., Information Systems Planning: What makes it succesful?, in Lawrence M. (ed.), Information Technology Towards 2000, Proceedings of the Australian Computer Conference, 1988

Treacy M.E., An Empirical Examination of a Causal Model of User Information Satisfaction, Center of Information Systems Research, Sloan School of Management, Massachusetts of Technology, 1985

APPENDIX: List of the variables used in this article and their means and standard deviations

Variable	Mean	Standard deviation
USER PARTICIPATION		
Extent of representation (number of users participating/ total number of users * 100)	36.0	30.5
Users' workload (percent of the total workload)	23.8	20.9
Adequacy of participation (1-7)	5.42	1.55
Users' commitment (1-7)	5.04	1.67
Users' DP skills (1-7)	4.02	1.48
Users' knowledge of supported business (1-7)	5.68	1.33
Users' communications skills (1-7)	4.93	1.35
Users' ability to specify requirements (1-7)	4.72	1.41
Influence of user participation (1-7)	5.52	1.16
SYSTEM ANALYSTS' PARTICIPATION		
Extent of representation (number of system analysts/ total number of project members * 100)	56.4	22.1
System analysts' workload (percent of total workload)	39.0	29.4
Adequacy of participation (1-7)	4.27	1.63
System analysts' commitment (1-7)	4.76	1.73
System analysts' DP skills (1-7)	4.89	1.59
System analysts' knowledge of supported business (1-7)	4.46	1.62
System analysts' communications skills (1-7)	4.74	1.55
System analysts' ability to specify requirements (1-7)	4.66	1.52

APPENDIX: continuing

Variable	Mean	Standard deviation
PROCESS VARIABLES		
Time overrun (percent)	45.6	57.5
Budget overrun (percent)	16.6	26.5
Success of requirements specification phase (1-7)	5.10	1.18
Success of logical design phase (1-7)	5.10	.99
Success of physical design phase (1-7)	4.89	1.41
Success of implementation phase (1-7)	5.08	1.18
Success of use phase (1-7)	5.40	.99
QUALITY OF AN IS		
<u>UIS-attributes:</u>		
Completeness of output information (1-7)	5.48	.99
Relevancy of output information (1-7)	5.13	1.07
Precision of output information (1-7)	5.39	1.48
Accuracy of output information (1-7)	5.32	1.36
Reliability of output information (1-7)	5.39	1.25
<u>User interface attributes:</u>		
Ease of use (1-7)	4.95	1.28
User friendliness (1-7)	5.08	1.38
<u>Maintainability atributes:</u>		
Error correction (1-7)	4.15	1.31
Implementation of changes (1-7)	4.26	1.51
Adaptivity to new requirements (1-7)	4.35	1.55
IMPACTS OF AN IS		
Goals achieved (1-7)	5.21	1.11
Improvements (1-7)	4.62	1.07
Profitability (1-7)	3.89	1.35
Usage in relation to goals (1-7)	5.45	1.16

SYSTEM DESIGN METHODS AS CREATIVITY "KILLERS"

by

Erik Stolterman
Institute of Information Processing
University of Umeå
901 87 Umeå
Sweden

Abstract: A lot of questions arise when creativity is discussed in combination with the concept of method. In this paper the meaning and place of creativity in the system design process is discussed. There are some basic theoretical assumptions about the "nature" of the design process that very strongly determine the possibilities of a creative process. Many methods today are based on a view of the design process that is very hard to combine with creativity. These methods could be seen as *creativity "killers"*. Two ways to understand the design process is discussed, design as *problem-solving* and design as *invention*. The invention view is elaborated as an better alternative, since it stimulates a continuing debate on subjects such as aesthetics, ethics and ideology. These are necessary concepts if we want to succed in our efforts to enhance creativity in the design process.

Keywords: Design process, creativity, methods, problem-solving, invention

Introduction

There is an increasing interest in the relation between creativity and system design. A common view is that creativity plays an important role in system design. This view is seldom questioned. But why is creativity necessary? Does the search for creativity indicate that system design today is performed without that "magic" component? Is it possible to design systems without creativity? If it is, then why should we discuss such a fuzzy concept as creativity? In order to sort this out, we have to ask some questions: What is the advantage that creativity provides? Where and how does creativity appear in the design process? What is possible to achieve and what are the goals with a better "use" of creativity?

All these questions arise when creativity is discussed in combination with the concept of method. The concept of creativity has, as a general concept, almost only positive connotations, which might very easily lead to an unquestioned and uncritical acceptance of everything that promises more creativity. In this paper I will examine the meaning and place of creativity in the system design process. I will show that creativity is not something, a "thing", that is possible to add to any design process or any design method. There are some basic theoretical assumptions about the "nature" of the design process that very strongly determine the possibilities of a creative process. I will argue that many of the methods today have a basic view of the design process that is very hard to combine with creativity. These methods could be seen as *creativity "killers"*.

I think that methods in general have the effect that they often create obstacles to a creative way of working. If we want to encourage creativity, the question is less how to increase the amount of creativity in the process, as how to remove the obstacles. This unwanted(?) situation, with methods that don't support creativity, is a result of some major misunderstandings of the "nature" of the design process. But it is sometimes a result of a very conscious strive. In many cases design methods have been designed *not to support creativity*. This is the result when the design process is viewed as *problem-solving* and not as *invention*. In order to make this more explicit, I will discuss some ideas on the "nature" of the design process. And I will try to present a possible way to reach a design process where creativity and invention have a stronger position than in today's methods.

Why do we need creativity?

The concept of creativity has very positive connotations. Almost in every context where the concept is used, it symbolizes something good. A good author, painter, scientist, teacher etc. works in a creative way. But what does this mean? What is the goal, what is the award of being creative? Is it that the solution, the product, of the process has some specific and wanted characteristics? It is not difficult to find a lot of qualities connected with creativity. One is *originality*. If a designer is creative, it is likely that the result of his work has some kind of originality. Another quality shows when the product in some way reflects the character of the designer. We get a product with *personality*. There are more, such as *beauty*, *style, fashion, smartness*.

Is this what we want when talking positively about creativity? Do we need information systems with qualities such as originality and personality? If so, where is the on-going research trying to reach that goal? Unfortunately, this kind of research seems to be very difficult to find. Instead, these qualities are the very opposite to the ideals that guide research on system design methods today. A lot of money and work is dedicated to finding methods that assure an acceptable result. The main interest seems to be to avoid bad solutions, not to make excellent ones. The aim of research is in many ways focused on the problem of getting the design process under control. This, by developing detailed specifications on how to perform the process. In this search for control, creativity becomes a threat. It is difficult to maintain control over a process where unforeseeable actions caused by creativity might occur. So, creativity is not wanted by everybody[1].

If we want to increase creativity, we meet a second problem. Is it possible to measure the amount of creativity involved in a specific process? In the design literature two concepts are sometimes used to describe the problem of measuring the effects of creativity: *goodness* and *correctness*. Goodness is defined in many ways, for example: goodness is a quality of a design that goes beyond correctness and could possibly become a *classic* design (Rouse, & Boff, 1987). That is, creativity is the thing that makes the designer able to "add" goodness to a correct solution. A correct solution, it is assumed, is possible to achieve without any help of the magic of creativity. So, if we look at the design process only as a way to reach a correct solution, then creativity is not needed. Viewed this way, creativity becomes some kind of luxury. It also becomes something dangerous, because it makes the designer unpredictable and difficult to control with a method. This dangerous aspect of creativity could lead to very concrete problems. In some of the CASE -literature the strive to control the process is very obvious and explicit. One goal, for instance, is to control the process in such a way that it is possible, whenever wanted, to exchange one system designer by another (Martin, & McClure, 1988), even in the middle of the process. The designer becomes an operator, with no need of creativity.

[1]I will at the end return to this fundamental choice of what kind of design process we need. The choices could in a more colourful way be described as the "routine"-process and the "adventure"-process, with the concepts of M. Berman (Berman, 1983).

Is this what is missing in todays system design methods, a strive for goodness? Are the methods totally focused on correct solutions? I think that if we want new solutions, this is in many ways the right answer. We need new solutions, we need "dangerous" system designers. We need creativity and we need a discussion of goodness. Goodness as a way to measure systems on a far broader scale than is usually the case today. We need some kind of *language*. A language that makes it possible to discuss this goodness. It is not anything mysterious or strange. In other areas this is often called *aesthetics* and *ethics*. Sometimes expressed in some kind of *ideology*.

Why most methods don't stimulate creativity.

It is perhaps true that there is a consensus about the need of creativity in the design process (although the reasons may differ), but there is no consensus on where in the process this creativity is needed.The goal of the research within our field is often to develop some kind of method. A method is seldom complete, neither in range nor in depth. The creators of a method has some idea of what is the most important activities in the design process, i.e. where in the design process a method is most needed. Among all the different methods existing, with different ideas of what a method should concentrate on, it is possible to distinguish several approaches. Approaches or views on how one best should support the design process. This kind of approaches has been described by many authors, in many ways (i.e Bansler 1987). This kind of classification is often based on the thought that different design methods share some fundamental ideas. I have, instead, categorized approaches according to what could be seen as their main interest (or research object) in their strive to develop methods. My intention is not to give a complete picture of all existing "schools", but to show that despite the very obvious differences between these approaches, all of them have problem to incorporate creativity in the process. And, for the same reason.The three approaches in my categorization are:

1. The method as a question of *formalisms*
2. The method as a question of *organization*
3. The method as a question of *philosophy*

Within the first approach, the interest is focused on how to improve and increase the formal parts of the system design process, and the purpose is to maintain control and to get an acceptable result in a secure way. In order to achieve this, it becomes essential to reduce creativity, since creativity is very difficult to control and to incorporate into formalisms. The second approach, the idea to organize, is often guided by goals such as democracy or co-operation or negotiation. These goals are not directly connected with creativity and radical solutions. On the contrary, they might lead to compromises.

These two approaches are the most common. It is, though, possible to recognize one more: the philosophic approach. Research within this approach seldom results in a traditional method. Its purpose is more often an attempt to explain other approaches or to establish some kind of theory on system design. A philosophic approach often begins by stating the "nature" (the ontology) of system design. Usually stipulating the meaning of concepts such as organization, data, information and knowledge. Continuing by defining the role of the designer, the user, the method, etc. Philosophic approaches have an opportunity to choose radical views on creativity. Unfortunately, this is seldom done. Instead they end up with some vague way of approaching a task, not explicitly stating the importance of a creative design process.

In none of these three approaches there is a "natural" place for creativity. The concept is seldom mentioned, and is not discussed as a crucial part of the design process. I think that this lack of interest in creativity is a result of the inability within these approaches to focus on the "real" design process. They provide knowledge that in several ways are both necessary and important, but they miss the most important aspect of design. And this by neglecting the importance of the origin of creativity.

Where is the creativity?

Creativity is probably one of the most basic human activities. Creativity occurs in the head of people. This is perhaps a very obvious fact, but it is crucial. Because it means that it it not possible to make creativity appear within a method, without a designer. The solution of the design process is always a result of a human creative process. And in that process, the designer himself is the most important factor. The result is not possible to deduce, neither is it a product of a certain kind of organization. The result has to be invented by somebody[1]. If we have a desire for more creative solutions, we have to concentrate on the designer. One thing to remember is that every formalism, organization and philosophy is something that imposes different obstacles and possibilities on the designer. The problem today is that most methods provide more obstacles than possibilities. They "kill" creativity. This "killing" is not altogether a result of the number of obstacles. It is more a question of what kind of obstacles. As with painters or composers, the right kind of restrictions (obstacles) might even stimulate creativity. For example, the painters who chooses to paint only with one or two colours, or the composer who chooses to compose in some extremely strict musical form.

[1] This is argued in (Mathiassen, 1988)

It seems that the three approaches, mentioned above, all seem to result in a aggregation of the wrong kind of restrictions, at least if we want to stimulate creativity. Despite obvious differences between the approaches, they share a basic view on the design process. A view of the process that leads to the wrong kind of restrictions. The design process is viewed as *problem-solving*. With this view, it is very difficult to stimulate (and increase the amount of) creativity in the process. In order to explain this dilemma, we have to examine the "nature" of the design process.

The design process

It is impossible in this short paper to discuss the nature of the design process in all its details. I will therefore concentrate on two ways of viewing a design process[1]. They correspond to the already mentioned concepts of correctness and goodness. These two ways of viewing the design process are to view it as *problem-solving* and to view it as *invention*[2]. I will discuss some aspects of the differences between these views. In the real world we seldom find these clean cuts between different views[3]. The views are, of course, idealized and described as extremes. Even if I do find the problem-solving view dominant in the area of systems design, I seldom meet any real hard "true believers". The invention view is perhaps even more rare. It may be seen as a short and idealized version of my own view of the nature of the design process.

The problem-solving view

One way to describe the difference between the problem-solving view and the invention view is by the concepts of *problem* versus *ideal*. Within the problem-solving view, problem-solving is the activity that has to be performed in order to solve a problem. A problem is something "given" or something that has to be "found". This is based on the assumption that problems *exists*. As a system designer you are either given a problem or you have to search for it. With good methods of analysis, it is always possible to find the correct problem. Within this view, a problem is something that has a solution. Problem-solving is to "find" that solution. It "exists" in the same way as the problem. A method is considered to be a rational step-by-step action to "reach" a solution. A solution is a result

[1]Someone might ask why it is so important to study different views of a process. A view is neither right or wrong, so what is the point? The point is, and it is very important, that a designers view of the process affects his own possible ways to perform that process. This is very well described in the book "Thinking on paper" (Howard, 1986), which is about writing."What you think writing is (or isn't) can profoundly affect how you do it" (page 24).

[2] These two views (and similar ones) could be found in the literature under a variety of names. For example as the *analytical view* versus the *artistic view*. (Rouse, 1987), or as the *depictive view* versus the *constructive view* (Forsgren, 1988).

[3]It is very difficult to find any research on how real designers view the design process, and how they view the relation between creativity and methods. We are at the moment planning a research project with the purpose to make such an investigation (Stolterman, 1989).

of the nature of the problem and the quality of the method. Since problems exist, they are possible to classify and distinguish by their nature. For instance, as easy and straight-forward or as complex and hard to solve. Since the problem exist and the problem has a solution, the method is the "way" between them. A good method is reliable and "reaches" the solution in an smooth and economic way. With this view, the problem becomes a very important "object". When the problem is found, simply apply the method, and the solution will follow.

Within the problem-solving view, the purpose with a method is to establish *security*. There is a "path" from a problem to the solution. The method is the right path. The method is also so well specified that the executor of the method doesn't have to worry about how to get to the result. If he just follows the method, it is no problem. The assurance is in the method.

With the problem-solving view there are some factors in the design process that becomes more important. These are: to find the problem, the nature of the problem, to apply right method, to have a correct method. There is very little interest in things like the designer, quality and characteristics of the solution (since correctness is enough). The research based on this view concentrates on "tools" for analysis and deduction, and methods based on these tools.

Even though the process is viewed as a matter of careful analysis and correct deduction and not as art, creativity is sometimes wanted. There is an understanding that even in very formal ways of problem-solving, there is sometimes a need of a good guess. Heuristics is needed not to improve the quality of the result, but in order to get economy in the process. Since the skill to make a good guess, seems to be a result of creative thinking, creativity becomes something that improves the speed of the process but not the quality. Creativity is not viewed as anything that contributes by adding new characteristics or dimensions to the result of the design process, but is rather seen as a component in the economy of the process.

The invention view

Within this view, problems and solutions are *invented*. Problems do not "exist". The problems are, in the same way as the solutions, to be invented. The solutions are not possible to qualify as correct or not. They have to be measured in terms of good and bad. This measuring has to be done in relation to the context where the design takes place. Since there is nothing that guarantees the result of a design process, the whole process depends on the designer. The designer must know what kind of result he wants to accomplish.

This view is to a much lesser extent guided by the assumed laws of nature as is the case with problem-solving. The designer can not rely on strict methods of analysis and deduction. Not any action, not any decision he makes, is determined by "the nature of the problem" or by the laws of deduction, instead they are results of choices made by the designer. The design process is, from the beginning to the end, an on-going process of choices. A designer need to have something guiding him in these choices, such as ideas of what is possible to do and ideals of what result he wants to achieve. This moves the interest away from correctness and towards goodness. Away from true or false and towards values and ideals. A designer need to have a well elaborated sense of aesthetics and ethics. This view puts a lot of pressure on the designer. The purpose of a method becomes to support and enhance the designers ability to be creative, since this is the most important factor in every design process. If we accept the invention view, we need some explanation and understanding of that ability? What makes a designer creative? Is it the method he uses, is it the techniques or what?

An example. To interpret and perform a piece of music is an act of design and is usually considered as creative work. The concepts of goodness and correctness are possible to apply in this situation. The music piece may be regarded as having one correct interpretation. The task of the musician is to interpret the notes and try to create the sounds that the composer had in mind. The musician has to practice in order to improve his interpretation of the piece. An interesting question now arises: how does the musician know when his interpretation is improving? Well, he could perhaps try to find out a (the) correct way to "read" the composer's symbols. A correct reading should, of course, result in a correct interpretation. I do not think that this "problem-solving" -like music-making is very fruitful. With music it is fairly easy to understand that a good result is not possible to obtain only by applying a correct technique. The musician must have the ability to recognize when it sounds good. At least to know when it sounds better or worse. He needs ideals about how the music should be. Otherwise, he has no possibility to know when he is improving. It is not enough to be able to use and master a technique. First of all it is necessary to have an elaborated sense of the goodness of the result.

Here is perhaps a key to understand creativity. Creativity is something that is stimulated when the designer knows what he wants to achieve. Not in terms of a precise description of the product, but as a well developed sense of aesthetics, collected during critical studies of design ideas, design ideals and of the work and work process of other designers. With this developed sense, he becomes creative and invents ways and means to accomplish his ideals. A good musician that cannot reach his ideals with the ordinary "tools" of playing invents new ways to play. He becomes creative. This is in a way a paradox. To stimulate creativity in the design process, it is necessary to focus the interest on the result (in the meaning mentioned above) and on the designer, and not on the process. If the process itself is regarded as the main "object", and there is no acting subject (designer) or discussion about ideals, the search for a new way to guide the design process could easily end up in another step-by-step method, without any support or explanations to the role and nature of creativity. This is what happened to some of the so-called process oriented research. It is not possible to "use" creativity as a component in the design process, and at the same time in detail prescribe what kind of product this creativity should end up with. This would not be real creativity.

With the invention view, research on system design would have to extend its area of interest. The designer should be placed in focus. And there would be a greater awareness of the need to establish and develop a conscious sense of aesthetics and ethics.

My purpose by presenting two opposite views of the design process, is to create a better understanding of the process where creativity is meant to appear. Since creativity is such a "fuzzy" concept, it is important not only to understand what creativity is about, but also to examine under what circumstances it is possible to stimulate creativity. My hypothesis was that the view of the design process very strongly determines the possibilities for creativity to grow. My conclusion is that a problem-solving view radically restricts that possibility. With the invention view, though, it seems to be the opposite. Within this view, philosophic assumptions about the nature of design and of creativity are not only easy to combine, they are indispensable to each other.

Conclusions

One conclusion is that the major part of system design methods is dominated by the problem-solving view. When the design process is viewed this way it becomes very difficult to incorporate creativity in that process. There are three reasons. First, it is very difficult since creativity is a basic human activity, performed by designers. And this view underestimates the role of the designer. Second, the possible demand for creativity has not the purpose to reach qualities such as personality and originality. If there is such a demand for creativity, it is based on economy. And third, and most important, the relation between the problem-solving view and creativity is difficult to establish, since almost every "tool" and technique used within this view is dedicated to some kind of specification tasks. Tools and techniques used in order to describe organizations, to specify conceptual schemas, to specify all kinds of information and data flows and system structures. Since these tools are implied to work in a deductive and formal way, they are not by "nature" in need of a creative designer. This is a bad mistake. Specification is not some kind of passive describing. Specification is always design, it always needs creative choices. It is not possible to make conceptual schemas without someone deciding what should be regarded as "object" or "relation", etc. It is possible to claim that the problem-solving view tries to hide (or "kill") the creative dimension of all design work, by using concepts such as specification and analysis instead of invention and choice.

So, if we leave this problem-solving view, and enter the world of invention, we get another view on creativity. Creativity becomes something which removes the criteria *correctness* and introduces *goodness*. This change of criteria also leads to a different purpose with the research on system design. It should more concentrate on *aesthetics* and *ethics* of the result of a design. We must create ideals that may guide the designer in his work. It is an on-going, never ended, discussion on these ideals that, at the moment, is the most important research within system design. It seems that a shift in interest away from the outer form of the design process (organization etc.) and towards a greater interest in the values and qualities of the "product" is needed. And most of all, if we want creativity to be an important part of system design, we also have to accept a new view of the designer. He must be seen as an inventor and not as a not responsible operator of a method. With creative design, where choices are the important design moments, responsibility follows. A choice is always connected with responsibility. A "correct" solution is not in the same way someone's responsibility.

If we continue to focus on questions of organization or formalisms, it is not possible (at least not easily) to achieve a creative and radical system design. The approach of philosophy is in fact the one that might provide us with a new view of the design process and of the designer. With a philosophy that states the importance of the designer, and with discussions on *aesthetics* and *ethics* of the "product", we could establish a basis for a better support of a creative design process.

This leaves us with a choice, perhaps a choice that could be described as either design as "adventure" or design as "routine" (concepts borrowed from Berman 1983). The routine way (the problem-solving view) offers security and standard solutions. The adventure way (the invention view) offers astonishing results, solutions never seen before, radical new ideas. Ideas and products that sometimes will be very good, but (since there is no guarantee) sometimes bad or even dangerous. The final conclusion must be that there is in fact no real choice between these two ways. The security offered by the problem-solving view is no more than an illusion. The danger lies not in creative and radical solutions, the real danger is the belief and trust in a design process guided by "invisible" laws of nature, guided by the "existence" of problems, correct and precise methods, and a dangerous trust on the idea that a design process could be done without a totally responsible inventor. Our task is to criticize a view of that kind, and to put forward a view not based on laws of any kind, but based on an continuing debate on subjects such as aesthetics, ethics and ideology.

References

Bansler, J. (1987). *Skandinavisk systemudvikling - teori og historie* . Studentlitteratur.

Berman, M. (1983). *All that is solid melts into air: the experience of modernity.* London: Verso Editions.

Forsgren, O. (1988). *Samskapande datortillämpningar - En systemteoretisk ansats för lösning av vissa förändringsproblem vid administrativ datoranvändning.* (Thesis). Institute of Information processing/ADP. University of Umeå.

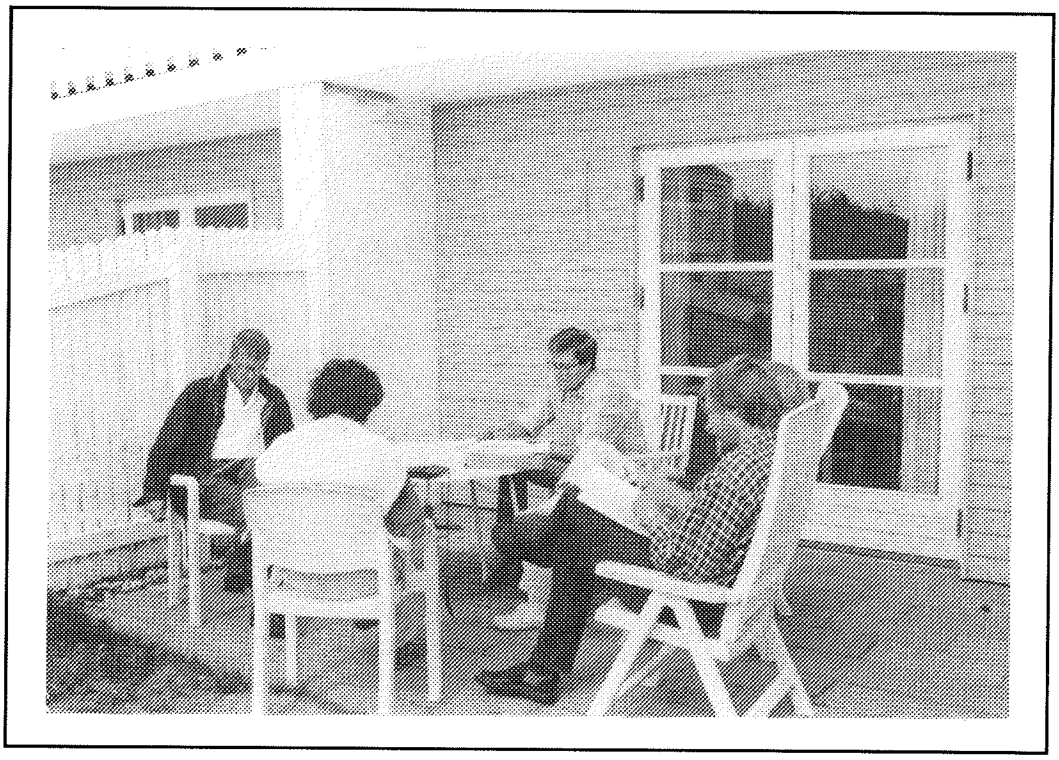
Howard, V. A., & Barton, J. H. (1986). *Thinking on paper* . New York: Quill.

Martin, J., & McClure, C. (1988). *Structured techniques the basis for CASE* . London: Prentice Hall.

Mathiassen, L., & Munk-Madsen, A. (1988). *Myths and reality in software development. Institute of electronic systems,* Aalborg University Center. (Draft version). (To be presented at the 11th International Conference of Software Egineering, Pittsburgh, Pennsylvania, USA, May 15-18 1989)

Rouse, W. B., & Boff, K. R. (1987). *System design, behavioral perspectives on designers, tools, and organizations* . New York: North-Holland.

Stolterman, E. (1989). *Metoder och datorstöd för systemutveckling* . (Projektförslag - utkast). Inst. för Informationsbehandling, Umeå universitet.



Group work: Bo Dahlbom, Lucy Suchman, Timo Vaara
and Kaj Grønbæk

System Maintenance and Organisational Change

Pål Sørgaard Åbo Akademi

Markku Nurminen University of Jyväskylä

Ulf Forsman City of Turku

Abstract: There are many indications that maintenance of computer-based information systems is critical. It consumes many resources and it is highly necessary in organisations which have made their information systems a central part of their operations.

We can also observe that organisations constantly change. Besides the permanent need to adjust to changes in the organisation's environment, there are tendencies towards decentralisation and to move away from classical hierarchy towards flatter organisations (e.g., cooperative work).

This raises the question: how and to what extent can the computer-based information systems become obstacles to organisational change? How is this phenomenon related to the difficulties in maintaining and changing the current systems?

We propose that research is made to investigate this issue. Such a research effort can be justified by existing quantitative results showing that most of the maintenance work is done due to changed user requirements. We want to study this problem in detail, and we therefore propose a research project with four phases: preliminary investigations (interviews and literature study), in-depth analysis of the situation in one or two case-organisations, formulation of change strategies, and implementation of one or more of the change strategies. We do in other words propose case-based, qualitative research in an area which primarily has been researched by quantitative techniques.

Introduction

This paper describes a planned research project in system maintenance and organisational change.

We will introduce the problem to be studied by the project by referring to a number of observations:

- In daily life the ordinary citizen experiences that many companies and public authorities provide an extremely inflexible customer service.

- Computer use often causes stress to the users.
- New system development projects have difficulties in getting resources because the developers have to attend to urgent problems in their "old" systems.
- Maintenance of computer systems consumes an alarming amount of resources.
- Software maintenance is considered low-status work in dp departments.
- Statement from a system developer: "It is much easier to develop a new system from scratch than to replace an existing one".
- Statement from a manager: "In our attempts to change the structure of our organisation we feel that our computer systems keep us locked up like in a jail".

These observations are not unrelated. Together they show us a picture of growing problems with the existing computer-based information systems. The software crisis has become a maintenance crisis. The problem is severe. An increasing amount of resources are consumed by system maintenance, but the problem does not seem to disappear. Bad systems is a severe problem for the work-environment. Nobody knows how many resources are consumed due to lacking maintenance of inappropriate systems. These are costs arising from the inexpediencies of using such systems, from the creation of hidden procedures to work around the system, etc.

The observations above form a problem which can be formulated and understood in many different ways. The next section of this paper is devoted to a discussion of different ways of formulating this problem. These formulations are divided in two groups; a group of technical formulations and a group of what we call extended formulations. We believe that there is a piece of truth in all of the formulations presented, but in this project we will mainly focus on the extended formulations. We do this because we are convinced that the problem is not only a technical problem and because we are interested in studying the relationship between the computer-based information system and the organisation. We are especially interested in *the situation where the computer-based information system becomes an obstacle to change in the organisation.*

The problem area outlined above is extremely large, even if we settle for a limited set of formulations of the problem. In order to do research we need to make a further delimitation of the problem. This is done in the third section. Finally there is a large section on our research strategy.

Problem formulations

The problem we are discussing here can be formulated in different ways: each formulation reflecting a way of seeing the problem. Behind a formulation there are implicit assumptions about the problem domain, for example about the selection of issues to include in an analysis of the problem. Accordingly, there is an implicit view of the causes of the problem, with a related promise of a solution: if we remove the obstacles, then we will succeed. The formulations are, in other words, different explanation strategies and many of them may be used and have been used as a point of departure for research activities. We believe that there is a piece of truth in all of the formulations below, and consequently, that there is a wide array of legitimate approaches to the solution of the problem.

Technical formulations

First we will consider a set of technical formulations of the problem. The formulations are technical in the sense that they focus on the computer system as a technical construct or that they focus on the technical competence of the system developers:

1. The development environments (e.g., the programming tools and the languages) are inappropriate. This results in extreme inefficiency in software maintenance.

2. The specification of the system should be more precise, preferably by the introduction of some formal specification technique. The programs should be verified and automatic program provers should be used if possible. The system should in principle be a provably correct implementation of the specification.

3. The software is inappropriately documented. This makes it hard to change the software and, worse, it increases the risk that changes are made which violate the assumptions made when the software was written. The problem is the lack of documentation method or discipline to use it.

4. The software is inappropriately written and thus hard to maintain. Software should be constructed in a modular way so that changed requirements do not cause ripple-effects [30, 32].

Our general argument against these formulations of the problem is that they do not recognise that a computer system in many situations only can be fully understood in a context where we also consider the way the system can be or is being used, the division between automated and manually performed tasks, and the way the cooperation between the users and the customer service is supported or disrupted by the system. The system might be technically perfect, but it might simply be doing the wrong thing!

We are especially sceptical to the first explanation. Not because we do not expect a productivity increase from new tools, but because the dp society has a tendency to overestimate the effect of new development environments. The present fad is CASE (computer-aided software engineering). We prefer to be sceptical. Until now the effects of improved environments have been eaten up by the volume of work or by the need to change the software.

Parnas has argued convincingly against the second formulation. Not because he is against formal specification, but because the development process cannot be performed in a strictly rational way (i.e. as a process making a strict separation between specification and implementation) nor exactly according to standards or methods [31]. It might, however, be desirable to document the system so that it appears as a logical consequence of a requirements specification.

The weakness of the third argument is that it is not very operational. It may degenerate to a critique of those who made the system. There are many undocumented systems which need to be maintained. Furthermore it appears that system documentation deteriorates as the system evolves. There may be something wrong in the way maintenance work is organised or in our conception of documentation.

Similarly pont four may degenerate to clever after-thoughts. Critical decisions should be encapsulated in separate modules, but it may be very hard to know what decisions are critical, especially if the system developers lack a thorough understanding of the system's subject area. During maintenance, however, it will gradually be possible to identify those factors which do change. It is therefore important that system maintenance is seen as a gradual improvement of the construction of the system, and not as a discipline of finding the cheapest way to achieve the desired change [37]. In this way we may also hope that maintenance can become more interesting than it traditionally is.

Extended Formulations

Below we will present a set of extended formulations of the problem. We will in other words describe the problem in terms of more concepts than just the technical ones.

We will primarily focus on organisational formulations:

1. System developers and the management tend to assume that system development is a technical task, whereas they ignore the personal, organisational, social, and cultural aspects of the development process and of the product. As a result the system does not fit with its intended area of use, a fact which often is clear from the beginning, but which is made more apparent when the discrepancies between the system and the working practices in the organisation start to grow. Here the problem is seen as a problem of *the traditional dominance of the technical view.* Technical competence needs to be mixed with other kinds of competence if we are to make good systems.

2. The computer-based information system relies on a series of unrealistic assumptions such as assumptions

   - about stability in the user organisation,
   - that people work according to given procedures,
   - that inconsistencies between reality and the information in the system do not arise (e.g., size of stock),

   and a series of implicit assumptions made by the system developers (analysts, programmers, etc.) when the system was made; assumptions which simply had to be made because the system development situation made it necessary to decide whether things should be in one way or the other. Here the problem is seen as a problem of *inappropriate analysis and design,* with the focus turned to the way we understand organisations, what assumptions we make, and which of those assumptions we actually build into the systems.

3. Systems are traditionally developed according to the needs of management. The interests of the workers (the users) are ignored or only considered superficially in the development process. The management's perspective on the organisation is severely restricted, one source of problems is that management often believes that the work is performed according to procedures laid down or approved by management, a second problem is that management tends to focus on its own need for information whereas it underestimates the needs for lateral communication in the organisation. Grudin argues that the intuitive decision-making by management breaks down in the field of computer-supported cooperative work [11]. In this way the problem is seen as a *political* problem (the system should be based on negotiated compromises between the interests of management and

the workers) and as a problem of the *direction of organisational communication* (management focuses on vertical communication, the workers on horizontal communication).

4. There is a contradiction between the structure of the organisation and the change processes in the environment in which the organisation is a part. This has, for example, been formulated by Mathiassen [21, p. 58]. The computer-based information system has to be compatible with the current structure of the organisation as well as with the permanently changing environment. In this way the problem is seen as a *manifestation* of an inherent contradiction in all organisations.

These formulations are more realistic than the purely technical formulations simply because they extend the set of relevant actors and the set of possible explanations. The first formulation emphasises this. It simply says that a part of the problem is that we have not been willing to accept that the problem is more than a technical problem.

The second formulation expresses the problem the way we see it as researchers in computer-based information systems. It is not enough to encapsulate critical assumptions, it is also important to avoid making too many of them. In this project we will experiment with strategies for doing this, for example the HIS-model [25] and seeing the system as material [35].

The third formulation is relevant, but in this project we will not focus on studying the conflict between workers and management. A specific reason for this is that many organisations now try to remove unnecessary control and hierarchy. These are changes where there is a partial overlap of interest between top management and the workers. (The middle management may, however, be squeezed.)

The fourth explanation is an expression of the way we see the problem as a general problem in organisational design. We hope that further insight in the problem, as seen with this formulation, will give us new ideas for the role of the computer-based information system in the organisation.

Problem delimitation

Above we have already done some of the problem delimitation, simply by stating the ways we will formulate the problem. We are interested in the situation where the computer-based information system becomes an obstacle to change in the organisation. This has received different formulations above. Here we continue the delimitation by discussing the problem as a maintenance problem and by presenting a tentative view on the relationship between the computer-based information system and the organisation.

We will concentrate on the situation where the organisation already has a system. In this sense we focus on maintenance. We will not, however, look at all kinds of maintenance. We are not interested in repairs due to low technical quality or changes of hardware and systems software. We do in other words exclude corrective and adaptive maintenance (terms used by Burton Swanson [39]). We are interested in the situation where the organisation is changing, and where this leads to changes in the required functionality of the system. We expect that our focus on maintenance also will give relevant results for the development of new systems, simply because a new system must be developed with future needs for change in mind. Here we are not arguing for blind 'flexibility'. We intend to study the nature of the required changes, hence we hope to get knowledge about the kind of changes to expect, and thus about what assumptions one ought to avoid or which need a proper encapsulation.

We have often observed and even participated in discussions about the relative importance of development or maintenance: the two major positions in these discussions being:

- It is wrong to focus on development of new systems when the real problem is that we are drowning in maintenance of the systems.
- It is wrong to focus on maintenance because the problem will not be solved until we find a better way of building the systems from the beginning.

This disagreement is artificial: Focusing only on development of new systems is not so relevant when the bulk of existing systems is increasing and almost every system development project is a modification or extension of an already existing system. Focusing only on maintenance is also meaningless because we need to ensure that we do not repeat the failures of yesterday in our new systems and because many systems are of a so poor quality that replacement is better than repair.

We can observe that development of new systems becomes more and more like maintenance because the new systems replace or supplement old ones and because of prototyping and evolutionary development. Maintenance becomes more like development of new systems when one goes for a serious reconstruction of the system. This means that we should stop seeing development and maintenance as two different kinds of work.

In this project, however, we will delimit ourselves to study maintenance work. We believe that it is in this setting we will be able to get the clearest picture of the problem we address. We will make our own empirical investigations of maintenance work since we want to get a picture of the total situation formed by technical problems, needs for changes, lack of resources, etc. We do thus delimit ourselves from purely theoretical approaches like the invention of techniques for incremental development of software and the development of theories on the impact of organisational change on the computer-based information systems.

Research strategy

The aims of the project are basically two: The first is to get an improved understanding of the problem, the second is to try to find solutions to the problem. These two aims are related since it does not make much sense to try to solve a problem one does not understand and since an attempt to solve the problem is a way to test the new understanding of it. Besides this we believe that increased insight in the nature and severity of the problem will create justification for solutions at various levels of radicality. There is, however, not much need to prove the existence of the problem one more time, this has been done by other researchers. Much motivation can be obtained from Staffan Persson's book "Så tuktas en dator" [34].

We are planning four phases in the project. These are:

- preliminary analysis
- in-depth analysis
- solution strategy
- change effort

The two first mainly aim at increasing our understanding of the problem, whereas the latter two mainly aim at finding a solution to the problem.

In the following we will describe our plans for the four phases. Our main research method is action research, but since there will be difference from phase to phase we will describe our research method for each phase. Some previous research will be reviewed where relevant. A more complete review of literature on maintenance will be produced [36].

Preliminary analysis

The research problem has been preliminarily formulated already above. It is, however, based more or less on our pretheoretical, intuitive understanding. To make it a part of scientific discussion we have to relate it to previous research on related topics and to create a more precise and relevant formulation. The main activity in this phase, however, will be to make empirical investigations in a small selection of organisations.

The aim of this phase is exploratory; to find new sides of the phenomenon studied. If this knowledge interest is omitted, there is a danger of a narrowly defined formulation with a poor practical relevance. Another aim is to reach a formulation which is operative enough to form the basis for the other phases of the project. Exploration must not become explosion.

Previous research

A situation belongs to our problem area if there are computer-based information systems which become an obstacle to change in the organisation. Improved understanding on such a phenomenon calls for studies of three subproblems: (1) What are the intended organisational changes? (2) In what way is the information system an obstacle to this change? And (3) Which part of the information system (or which design decision) is the cause of the problem? We may expect that in different situations there are different problem formulations and different solutions. In any case there is, however, in the background a notion of *the relationship between the organisation and the information system.* One of our first tasks is to articulate our notion on this. Such a formulation may also shed some light on the realistic and unrealistic assumptions about organisations which may have been baked into existing information systems.

The notion on the relationship between the organisation and the information system has been discussed in many contributions at a rather general level. One researcher is Kristo Ivanov, University of Umeå, who is studying the presuppositions behind the development of computer-based information systems. This work does especially address the issue of the role of the computer-based information system in the organisation and the theoretical aspects of building assumptions about the organisation into the system.

The research object of the Knowledge and Work project has been existing information systems. This fits very well with our present research problem. The theoretical frame of reference [26] spells out directly that organisational problems are genuinely social ones and they should be treated as such instead of trying to computerise them. The project group has documented in its case reports how the organisational coordination has been baked into the structure of the systems [12, 27] in a way which makes the coordination invisible for the participating users. The group has also reported on negative impacts of computer-based information systems on organisational coordination [23]. As a solution the group suggests a social interpretation of computer-based information systems, which emphasises the role of the human actor behind all, even computerised actions, and they also have made proposals for the reconstruction of computer-based information systems according to this suggestion [12].

Even if the existing computer-based information systems are in the main focus, we certainly can improve our understanding also by looking at the development of them. This may give an understanding about how the problems have been created. The studies on development also have a maintenance dimension by telling us what can be changed easily and what cannot. The MARS-project [1, 15], Aarhus University, made empirical studies of system development in four Danish companies. In the MARS-project attempts were made to change working practices in system development according to the problems which were identified. Even if the research problem was not the same as ours, the setting of the project has many factors which will be useful in our project as well.

The QED project, Aalborg University Center, addresses directly the question of what system developers think when they design. This is interesting for our project, because it may tell us something about the mechanism which incorporates various assumptions into the design of the systems [22].

Our problem is not, however, a system development problem. The main attention of this project will be paid to the restructuring of existing, old systems. The volume of research on maintenance is quite voluminous, see [36], but there is practically no literature on how systems create obstacles to organisational change and on how this relates to maintenance. We are convinced that it is too simplified to define the problem as one of the costs of development and maintenance of software libraries, like the SOLE project, Åbo Akademi, does [40]. This project may, however, provide us with a conclusion similar to our point of departure: there is a real need to construct software in a different way which is easy to maintain in a changing organisation.

Pasi Kuvaja, University of Oulu, has previously demonstrated that 4th generation languages do not result in the promised reduction in development costs [17]. Other similar findings have provoked a counter-argument: These new tools (4GL, CASE, etc.) should be used since they reduce the maintenance cost. This is Kuvaja's present research problem. If he again finds that the efficiency gains are only moderate we will see this as a justification of our approach, since it will confirm our assumption that the real problems in maintenance are not of a technical nature. It has, by the way, already been argued that fourth-generation languages actually will increase the problems in maintenance [6].

Empiry

The more precise formulation cannot be done adequately by only reading previous research reports. We will need concrete computer-based information systems and organisations. We therefore want to make preliminary empirical investigations in several organisations. The main aim with these investigations is to ensure that we focus on the right problems and choose relevant organisations and systems for investigation in the later phases of the project.

What we will do is to interview various key-persons (typically administrative managers, outspoken users, or dp managers) in a small selection of companies. We plan to make approximately 5–20 such interviews.

To the extent it is possible we will select the interviewees according to the following criteria: He or she should be a key actor who has realised the need for an organisational change, who has tried to make it happen, and who has failed in this, at least partly due to the computer-based information systems. Furthermore we will try to delimit ourselves to settings where the computer-based information system is of an adequate technical quality, where reasonable development tools are available, and where the system developers are reasonably competent. The purpose of these delimitations is that we want to focus on a situation where the computer-based information system is a bottleneck in the organisational change process, and where this is not due to trivial, although perhaps common, technical complications. It is, of course, unrealistic to ensure that our interviewees and their organisations fulfill these criteria, but we will use them in our selection of interviewees to the extent we find it possible. We will, however, put strong emphasis on these criteria in the selection of case-organisations for the later phases of the project. It should be noted that these restrictions imply a further delimitation of the scope of the project.

The research method in the empirical work in this phase is qualitative interviews. On method we refer to [33, 38]. Furthermore we plan to bring our interviewees together for a seminar where we will try to create group-discussions which can enrich their and our understanding of the problem.

The result of this phase is an answer to the question of what factors in existing computer-based information systems prevent the change of organisation and to the question about the mechanism through which these factors get their effect. The answers will be based both on theoretical work and on empirical findings. We have to be prepared for multiple answers. We may be able to synthesise the answers but it is also possible that what we here describe as our problem can be formulated better as several distinct problems. The empirical research also confirms (or weakens) the relevance of our problem formulation by the very existence (absence) of the problem. We also hope that we during this phase will be able to select our case-organisations. Our main selection criteria will be the criteria formulated above and the willingness of the organisation to participate in our research project. Finally, the plans for the subsequent phases of the project may be concretised on the basis of these results.

In-depth Analysis

The in-depth analysis will be performed in one or two case-organisations. We would prefer more cases, but our resources are limited. The plans for this and the later phases are not yet documented in detail. We will use the results from the preliminary analysis to make the final plans.

The results from the preliminary analysis will not be detailed enough for our purposes. We want to study the problem in a concrete situation and to be able to talk to those people and read those documents we find necessary. The aim of this in-depth analysis is to create a rich description of the situation in the organisation and to identify problems in the maintenance of an existing system and in performing organisational changes. The central object of the analysis will be the structure and functioning of the computer-based information system, and especially the nature of requests for changes in the system. We will also study how the computerised work-tasks are related to other tasks and how these tasks aggregate to jobs. Furthermore we will study how the work is coordinated. Such an analysis will give us a picture of what organisational features are baked into the system, be it with or without intention. We may be able to explain the requests for modification of the system with our observations of the work. Furthermore we may be able to explain how the system prevents changes in the organisation and also what characteristics of the system which cause this resistance to change.

Clearly such an intervention into an organisation can only be performed in close co-operation with people from the organisation. We plan to establish and participate in working-groups in the case-organisations. Candidates for participation in such groups are actors who have tried to make changes, system developers and maintainers, and users. In order to ensure active participation we will, if necessary, establish a separate working-group of users. The research work in the case-organisation will be performed in or by these working-groups. We realise, however, that the initiative will be on the researchers' side.

Although we are not aware of any analysis with the same objective, we find that the analysis method proposed here is very similar to that used in several other projects in the "Scandinavian" tradition, for example the MARS-project [1, 15] and the Knowledge and Work project [28, 29] where we have participated ourselves. Other highly similar projects are the Florence-project [4, 5, 3] and the Tik-Tak project [7].[1]

Solution strategy

It is practical to distinguish between finding the solution and the change process where the solution is applied. There are many factors promoting the potential success — or failure — during the implementation. It is therefore important that we can discuss and evaluate the suggested solution(s) also independently of the change process. Furthermore the formulation of solutions in a separate phase makes our ideas more explicit. The solution strategies can be research results worth publishing. The solution strategies can also serve as a set of hypotheses to be tested in the next phase, the change process.

The change strategy is dependent on the problematic situation, which of course varies according to the experienced need for the organisational change. We have to study the recent discussion on such changes. There are some key trends, such as customer-orientation, better service, individualised production, and hostility towards bureaucracy. From the employee side there are growing demands on job satisfaction, challenge and autonomy, to name a few. Such trends have also been subject to research.

The COOP programme at Aarhus University [2] works on computer-supported cooperative work, primarily support for cooperative design. Many of the organisational changes we foresee, and which may be prevented by the current computer-based information systems, are changes towards more use of cooperative work. Our project can in fact be seen as a project on how to avoid computer-disrupted cooperative work.

[1]We know there are other similar projects, and we do not try to mention all. Please do not get angry with us if we do not mention *your* project. If you think your project used a method or produced results of special relevance to us, please tell us!

There are interesting trends also in the public sector. In the city of Turku a project aiming at modernisation of the economic basis of the city, and, as a part of this, also aiming at flexible use of computers to achieve this end, has been performed from 1983 to 1988 [8, 9]. The NUDU project at the University of Umeå is concerned with the use of computers in providing neighbourhood services. This can provide interesting alternatives to the way computers are used traditionally in public services.

It is not certain that an existing system can be reconstructed in the desirable way. It may in other words be necessary to phase down the old system before the reconstruction can start. To handle this we need to find techniques for phasing systems down, or even phasing them out, in a *controlled* way. Here we can draw an analogy to many other technologies where the issue of how to get rid of the waste is an increasingly important question.

Another analogy can be drawn to economical history. One kind of technology can evolve through a series of stages or versions, but sooner or later this line of evolution ends. At this stage it may be necessary to remove the technology, and thus suffer a crisis or setback in efficiency, before a new line of development can be entered (Esben Sloth Andersen, Aalborg University Centre). This line of thought has given our project the inofficial name "Creative System Destruction".

Less radical solutions can be used in those cases where the old system essentially can be retained. The change needed may be a modification of the system so that it can be compatible with various alternative modes of organisation. The steps in this trajectory are of course accompanied by the corresponding steps in the organisation of work and skills of the users. Even if the entire system is not phased down in these strategies, it may be expected that we will have to take some steps backwards in order to be able to take the desired steps forwards.

One possible strategy is based on continuous change: permanently ongoing system development, applied by the AMIS-project, Lund [24]. Their action research project with strong user participation is based on an interesting frame of reference, and the idea of permanent development may be seen as another formulation of maintenance.

The IDEA-project at the Gothenburg School of Economics and Legal Science (Informationsstöd för Decentraliserat Ekonomiskt Ansvar) is a project which aims at supporting local cost control by semi-autonomous groups. It seems that local initiative and control are crucial for the development of such supporting tools [10, 14]. The basic strategy here may be characterised as "islands of computational autonomy", according to the similar strategy to automation in general. Instead of a total plan for automation, small islands are established, which then grow to greater ones. The Human-scale Information System of the Knowledge and Work project could very well lead to a rather similar strategy. Of course, the islands cannot be isolated, but the coexistence problem must be faced anyway.

The modernisation project of the economic basis of the city of Turku will not save the entire organisation at once. The stepwise strategy is, however, preceded by theoretical development work. The economy system has been defined as a language which is very strictly standardised, a property which paradoxically gives an extreme flexibility for defining different use situations and their organisations. Perhaps this system can be seen as a specialised development environment for economical systems.

There are also analogies between the modernisation project in the city of Turku and object-oriented system development. In object-oriented development like Jackson System Development [13] the system is a model which simulates the events in its referent system. This strategy has two strong sides: (1) It is made very explicit during the development process what parts of the real world (i.e. the referent system) which the system is concerned with. And (2) the functions of the system can easily be changed as long as they do not require changes to the model. Furthermore object-oriented programming is a technique which makes it possible to program in problem-specific terms. Or as put by Krogdahl and Olsen [16] (translation from [20]):

> The basic philosophy underlying object-oriented programming is to make the programs as far as possible reflect the part of the reality they are going to treat. It is then easier to understand and get an overview of what is described in the programs. The reason is that human beings from the outset are used to and trained in perception of what is going on in the real world. The closer it is possible to use this way of thinking in programming, the easier it is to write and understand programs.

It may sound strange when we simultaneously advocate two approaches which seem to be contradictory. The first is the central role of human subjects in using computer-based information systems (the social interpretation demanded by the Knowledge and Work project). The other is the separation of the tasks to be performed from the organisation of this work, a separation which may be based on object-orientation. The social interpretation regards all tasks as inherent parts of the job even when performed by means of computers. This brings the organisation of work to a constituent part of the computer-based systems, which seems to be exactly what should be avoided in the name of flexibility of maintenance in a changing organisational environment. This dilemma must and can be solved.

The result of this phase is the formulation of a change strategy which consists of two interrelated parts, an image of the future systems and their use on the one hand, and the image of the change process on the other.

Change process

The point of departure is one or more solution strategies developed in the previous phase. In that phase the interest of the researchers was dominant. The purpose was to create ideas and solutions without letting the practical implementation problems, such as power relationships, restrict the innovative process. In the change process our research becomes real action research. The aim is to make the change happen. This has to be done on the premises of the case-organisation.

Careful planning is vital. If the change process would fail, it is important to analyse the reasons for the failure. If the failure is caused by poor planning, it does not tell anything about the relevance of the solution strategies.

The change process is, in other words, a test of the solution strategies created on the previous phase. There are many possible outcomes of the test. The project may be stopped. It may be finished successfully, but the changed situation may — or may not — be as expected. Each outcome must be interpreted in order to evaluate the selected strategy. In any case the process must be documented carefully, if we want to give a justified interpretation of the outcome. The documentation has also another function. As a byproduct we can possibly report of interesting side-effects. Even if we cannot include it in the frame of the present project, a longitudinal follow-up could be very interesting.

It is important to note that a failure in the change process also is a research result. A failure will tell us more about the obstacles to change and hence more about the phenomenon we want to study: the interrelationship between computer-based information systems and organisation.

The generalisability of the results has the same restrictions as case studies and action research in general. But even one positive case has some power. It may function as a proof of existence as well as a proof of the possibility of solving a problem. Our results can be juxtaposed with results from other kinds of empirical research, like those of Lientz and Swanson which demonstrated that most maintenance is performed due to changes in the users' requirements [19, 18], or those from software engineering (for example [41, 42]). We may hopefully provide some detailed knowledge, and thus some raw-material for rich interpretations where other researchers provide statistically significant results or technically interesting observations.

References

[1] Niels Erik Andersen, Finn Kensing, Monika Lassen, Jette Lundin, Lars Mathiassen, Andreas Munk-Madsen, and Pål Søgaard. *Professionel systemudvikling.* Teknisk Forlag, København, 1986.

[2] Peter Bøgh Andersen et al. Research programme on computer support in cooperative design and communication. IR 70, Computer Science Department, Aarhus University, Århus, 1987.

[3] Gro Bjerknes and Tone Bratteteig. å implementere en idé — samarbeid og konstruksjon i florence-prosjektet. Florence-report 3, Institute of Informatics, University of Oslo, Oslo, September 1987.

[4] Gro Bjerknes et al. Gjensidig læring. Florence-report 1, Institute of Informatics, University of Oslo, Oslo, 1985.

[5] Gro Bjerknes et al. Læring som forutsetning for design. Florence-report 2, Institute of Informatics, University of Oslo, Oslo, 1986.

[6] Ned Chapin. Software maintenance with fourth-generation languages. *ACM Software Engineering Notes*, 9(1):41–42, January 1984.

[7] Jane Foged, Troels Jørgensen, Arne KJær, and Randi Markussen. *Håndbog om klubarbejde, edb-projekter og nye arbejdsformer.* HK Kommunal/Handels- og kontorfunktionærernes forbund, København, Juni 1987.

[8] Ulf Forsman. Menojäämien käsittelyjärjestelmä, yleissuunnitelma. Technical report, Accounting Office, City of Turku, Turku, September 1983.

[9] Ulf Forsman. Taloushallinnon rationalisointiprojektin loppuraportti. Technical report, Accounting Office, City of Turku, Turku, April 1988.

[10] Anders Grönlund and Sten Jönsson. Managing for cost improvement in automated production. IDEA arbetsrapport 14, School of Economics, University of Gothenburg, Göteborg, December 1988.

[11] Jonathan Grudin. Why CSCW applications fail: Problems in the design and evaluation of organizational interfaces. In *Proceedings of the Conference on Computer-Supported Cooperative Work, September 26–28, 1988, Portland, Oregon*, pages 85–93, New York, September 1988. ACM. ACM Order Number: 612880.

[12] Riitta Hellman. A fictitious his-reconstruct of an inventory information system. *Computers in Industry*, pages 301–310, 1989.

[13] Michael Jackson. *System Development.* Prentice-Hall, Englewood Cliffs, 1983.

[14] Sten Jönsson and Anders Grönlund. Life with a subcontractor: New technology and management accounting. *Accounting, Organizations and Society*, 13(5):512–532, 1988.

[15] Finn Kensing, Lars Mathiassen, and Andreas Munk-Madsen. Mars a research project on methods for systems development. MARS-report 1, Computer Science Department, Aarhus University, Århus, July 1984.

[16] Stein Krogdahl and Kai A. Olsen. Modulær og objekt orientert programmering. *DataTid*, (9), September 1986.

[17] Pasi Kuvaja. An experimental analysis of the selection and productivity of application generators for professional use. Research Papers Series A 10, Intitution of Information Processing Science, University of Oulu, August 1988.

[18] Bennet P. Lientz and E. Burton Swanson. Problems in application software maintenance. *Communications of the ACM*, 24(11):763–769, November 1981.

[19] Bennet P. Lientz, E. Burton Swanson, and G. E. Tompkins. Characteristics of application software maintenance. *Communications of the ACM*, 21(6):466–471, June 1978.

[20] Ole Lehrmann Madsen and Birger Møller-Pedersen. What object oriented programming may be — and what it does not have to be. In Stein Gjessing and Kristen Nygaard, editors, *ECOOP '88 European Conference on Object-Oriented Programming, Oslo, Norway, August 1988, Proceedings*, pages 1–20, Heidelberg, 1988. Lecture Notes in Computer Science 322, Springer Verlag.

[21] Lars Mathiassen. Systemudvikling og systemudviklingsmetode. PB 136, Computer Science Department, Aarhus University, Århus, 1981. Also DUE-report nr. 5.

[22] Lars Mathiassen. Kreativitet og disciplin i system design. *DATA*, 18(6):33–37, 1988.

[23] Jukka Niemelä. Informal organization, coordination of work and the use situation of ISs. In Pertti Järvinen, editor, *The Report of the 10th IRIS (Information Research seminar In Scandinavia) Seminar*, pages 535–553, Tampere, 1987. University of Tampere.

[24] Hans-Erik Nissen, Gunhild Sandström, and Göran Ekvall. Ansvars och medverkansformer i kontinuerlig systemutveckling. LU–ADB–R 88–3, Lunds Universitet, 1988.

[25] Markku I. Nurminen. Human-scale information systems. Report I 821, Institutt for informasjonsvitenskap, Universitetet i Bergen, Bergen, 1982.

[26] Markku I. Nurminen. *People or Computers: Three Ways of Looking at Information Systems.* Studentlitteratur, Lund, 1988.

[27] Markku I. Nurminen, Inger Eriksson, Anneli Finneman, Jukka Niemelä, and Marjo Snellman. Raportti työ- ja tietojärjestelmästä elintarviketehtaan lähettämössä. Report A 50, University of Turku, Computer and Information Science, Turku, December 1987.

[28] Markku I. Nurminen, Riitta Kalmi, Pirkko Karhu, and Jukka Niemelä. Social interpretation of information systems. In Monika Lassen and Lars Mathiassen, editors, *Report of the Eighth Scandinavian Research Seminar on Systemeering*, pages 172–191, Århus, 1985. Computer Science Department, Aarhus University.

[29] Markku I. Nurminen, Riitta Kalmi, Pirkko Karhu, and Jukka Niemelä. Use or development of information systems: Which is more fundamental? In Patrick Docherty, Klaus Fuchs-Kittowski, Paul Kolm, and Lars Mathiassen, editors, *System Design for Human Development and Productivity: Participation and Beyond*, pages 187–196, Amsterdam, 1987. North-Holland.

[30] David Lorge Parnas. On the criteria to be used in decomposing systems into modules. *Communications of the ACM*, 15(12):1053–1058, December 1972.

[31] David Lorge Parnas and Paul C. Clements. A rational design process: How and why to fake it. *IEEE Transactions on Software Engineering*, SE-12(2):251–257, February 1986. Minor correction in SE-12(8):874.

[32] David Lorge Parnas, Paul C. Clements, and David M. Weiss. The modular structure of complex systems. *IEEE Transactions on Software Engineering*, SE-11(3):259–266, March 1985.

[33] Michael Quinn Patton. *Qualitative Evaluation Methods.* Sage Publications, Beverly Hills, CA, 1980.

[34] Staffan Persson. *Så tuktas en dator.* Prisma, Stockholm, 1987.

[35] Pål Sørgaard. Object oriented programming and computerised shared material. In Stein Gjessing and Kristen Nygaard, editors, *ECOOP '88 European Conference on Object-Oriented Programming, Oslo, Norway, August 1988, Proceedings*, pages 319–334, Heidelberg, 1988. Lecture Notes in Computer Science 322, Springer Verlag. Also available as PB 247, Computer Science Department, Aarhus University, Århus, May 1988.

[36] Pål Sørgaard. An overview of research in maintenance. Report A 94, Department of Computer Science, Åbo Akademi, November 1989.

[37] Mickael J. Spier. Software malpractice — a distasteful experience. *Software—Practice and Experience*, 6(3):293–299, July-September 1976.

[38] Anselm L. Strauss. *Qualitative Analysis for Social Scientists.* Cambridge University Press, Cambridge, 1987.

[39] E. Burton Swanson. The dimensions of maintenance. In *Proceedings 2nd International Conference on Software Engineering, 13–15 October 1976, San Fracisco, California*, pages 492–497, Long Beach, CA, 1976. IEEE Computer Society. IEEE Catalog No. 76CH1125-4 C.

[40] Aimo Törn. Models of software accumulation. Working paper, Åbo Akademi, 1988. To appear in *The Journal of Systems and Software,* 1990.

[41] Stephen S. Yau and James S. Collofello. Some stability measures for software maintenance. *IEEE Transactions on Software Engineering*, SE-6(6):545–552, November 1980.

[42] Stephen S. Yau and James S. Collofello. Design stability measures for software maintenance. *IEEE Transactions on Software Engineering*, SE-11(9):849–856, September 1985.

CONTROLLED EXPERIMENTATION IN HUMAN-COMPUTER INTERACTION RESEARCH

Timo Vaara
Helsinki School of Economics
Runeberginkatu 14-16
00100 Helsinki
Finland

ABSTRACT

Controlled experiments are increasingly used in human-computer interaction (HCI) research. The trend indicates need to establish results which satisfy scientific criteria, such as precision, validity and generality, but are the experiments producing results which could be directly applied to enhance the quality of new user interfaces? We examine the role of controlled experiments as a source of advances in order to understand the major factors that contribute to improved user interface quality. Technological progress, proficient designers and the millions of users who choose their interface every day are recognised as the primary sources.

1. INTRODUCTION

Recent results of Mitchell and Welty (1988) show evidence of increasing experimentation in human-computer interaction research. Starting in the late 1960's (e.g. English et al. 1967), human factors researchers have carried out a rapidly expanding collection of experiments. Particularly in the 1980's the HCI-research has matured to a stage where the proportion of experimental studies has clearly increased.

The trend towards controlled experimentation is an apparent consequence of early research in human-computer interaction that was largely conceptual rather than empirical. Emerging conflicts between recommendations emanating from different sources (see e.g. Maguire 1982) and difficulties to operationalize the recommendations motivated the move towards empirical orientation. On a conceptual level it had been easy to propose desirable user interface features. Recommendations such as "the interface should be helpful" or "use the user's model", are fine ideas, but they are not particularly useful when faced with concrete design choices. What makes a system easy to use, consistent or self-descriptive? As a reaction against the early HCI-research that based its methods substantially on intuition and introspection, there have been calls for an increase in experimentation in HCI research. For example, Shneiderman (1987) argues that the early HCI-research suffered from lack of validity, generality and precision and states that "the techniques of controlled psychologically oriented experimentation can lead to a deeper understanding to the fundamental principles of human interaction with computers".

The appreciation of rigorously managed experimentation in HCI-research leans on the fact that scientific inquiry relies heavily on experimentation in many other sciences and it is often seen to provide a widely accepted and uniform method of testing hypotheses which relate independent and dependent variables (see Mitchell & Welty 1988). Shneiderman (1987) outlines the "reductionist" scientific method as follows: 1) lucid statement of a testable hypothesis, 2) manipulation of a small number of independent variables, 3) measurement of specific dependent variables, 4) careful selection and assignment of subjects, 5) control for biasing and 6) application of statistical tests. Traditionally, controlled experimentation has had a fundamental role in natural sciences, while on the other hand, it has been used seldom in social sciences due to the complexity and context dependency of human activity. The field of human-computer interaction stands interestingly in the middle of these two extremes, because human action is taking place in a relatively small and deterministically behaving context when a user operates within an interface. Consequently, controlled experimentation has appeared as an appealing method of inquiry in HCI-research in search for exactness and significance.

Although a controlled experiment has become a major method of inquiry, we can question weather the results of conducted experiments have had a major impact on the quality of user interfaces we are facing today. Practical software design is still based more on past habits or a desire to use the latest technology than on validated research results. Would the state-of-the-art in human-computer interaction be virtually different without research efforts devoted to controlled experiments?

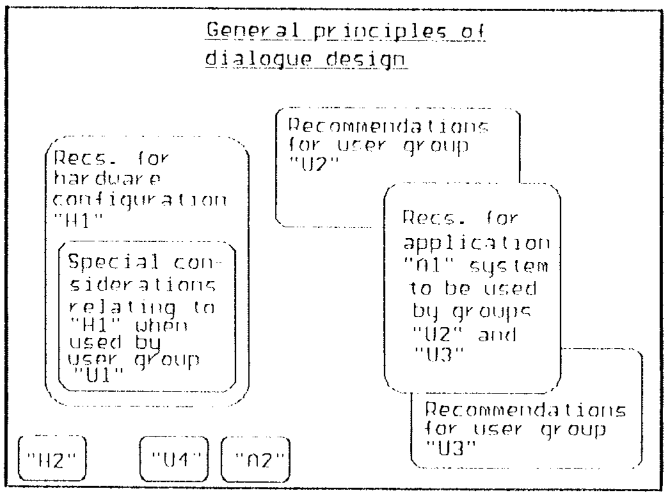
Prevailing situation in the field of HCI-research is idiosyncratic, because the quality of human-computer interaction has undoubtedly improved during the two decades of experimental research on human factors, even though it is difficult to identify the body of results that originate from reported experiments as the primary source of advances. Obtained empirical results - some even contradictory - have unfortunately revealed a considerable amount of dependence of situational factors (hereafter called as situationality), as often occurs in studies treating research problems that include human activity. Type of task, user's individual characteristics and hardware environment have been the major emerging situational factors. Also the trade-off nature of various design solutions has emerged. Conventional propositions by the writers (e.g. Moher & Schnider 1982) who have not been so pleased with the results the controlled experiments have produced, have included mainly means to increase the level of control in one form or another (random selection of subjects, bringing new factors under control etc.).

Controlled experimentation has clearly it's strengths and weaknesses as any other method of scientific inquiry. It has many possible pitfalls, for example, artificially produced, controlled and reproducible conditions may distort the situation so much that the results have no application or extremely good or poor performance by individuals may be overlooked due to the authoritative impact of statistics (Shneiderman 1987). However, our intention here is not to question the validity of the reductionist scientific method itself from a philosophical point of view, but to examine the role of controlled experiments as a source of advances in human-computer interaction in order to understand the major factors that contribute to improved user interface quality.

2. SITUATIONALITY IN HCI-RESEARCH

When a system designer is faced with the task of designing a user interface for an interactive computer system, he must make a wide variety of decisions. The bewildering amount of design choices makes it a demanding task to put together a combination best suited to meet the needs of human beings. It has been an ideal for the HCI-research community to provide system designers with precise and proven guidelines for resolving the myriad of design choices. There are many accounts of user interface design guidelines (e.g. Cheriton 1975, Gaines 1981, Gaines & Shaw 1983, Gardiner & Christie 1987), but it has been pointed out (Alty et al. 1986, Marchall et al. 1987) that it is somewhat problematic to employ the guidelines in practical software design. General context-free guidelines, such as "be consistent" or "allow the user to control the system", give little advice how such statements could be applied in specific situations, while on the other hand the more detail covered in a guideline the greater is the risk of reduced power in any given application and contention with other guidelines.

Maguire (1982) claimed in his evaluation of published recommendations that conflicts between guidelines being offered as general purpose advice are due to the tendency to base recommendations upon a limited field of experience. He suggested that such conflicts could be resolved by regarding each recommendation as an alternative strategy to be applied in a specific situation: "Without being placed into the context of a particular interactive environment, such information becomes devalued and apparent contradictions tend to arise." As a resolution to the problem of "conflicting" recommendations Maguire proposed the following framework within which current and future design information could be placed.


General principles of dialogue design
Recs. for hardware configuration "H1"
Special considerations relating to "H1" when used by user group "U1"
Recommendations for user group "U2"
Recs. for application "A1" system to be used by groups "U2" and "U3"
Recommendations for user group "U3"
"H2"
"U4"
"A2"

Maguire 1982

According to Maguire "the outer frame represents general principles of design which (if any such principles exist) would apply in all circumstances. Areas within this frame represent sets of recommendations relating to a particular domain of interaction, such as a particular application, group of users or type of hardware equipment." Maguire is thus suggesting a situational approach to structure the knowledge of user interface design. In the framework applications, users and technology are seen as the major contingent factors.

Anyone who is familiar with the contemporary HCI-research would hardly disagree with the basic idea of situationality that is underlying Maguire's frame of thought. In fact, the analysis of different contingent factors is an essential part of many user interface studies. However, before unquestionably adopting the situational approach as a basis for our research inquiries, it is worth while to consider two critical aspects that underlie the approach.

First, let us consider a situation where we have defined a set of contingent factors that introduce situationality into our universe of discourse. After conducting some successful experiments we have been able to produce some justified claims (in relation to the contingent variables), but according to the very idea of situationality, there is bound to appear new conflicting claims that introduce situationality in a more detailed level. Are we just converting situationality in the large into situationality in the small? Do we have any guarantees that we will ever reach a level where the situationality has been broken into it's atomic particles? The above framework by Maguire (1982) rests implicitly on the belief that such a level exists.

Second, the very idea of situationality claims that the effect of contingent variables on the subject under study is determined by combinations of contingent variables rather than variables as single entities (described in the above framework with an overlapping). Let us consider a situation where we have many contingent factors and the subject under study is highly sensitive to them all. Do we have any guarantees that we are not facing a situation where all appropriate design solutions are unique since every condition is determined by a novel combination of attributes? Are we in danger of looking for general rules in a n-dimensional space where every single point should be studied as a discrete entity in itself? The way how the above framework by Maguire (1982) is projected onto a two dimensional space is illusive. A three dimensional illustration would give a more realistic depiction in the case of three contingent factors.

The preceding two critical aspects of situationality are complementary views to the same problem. Bodker (1987) has nicely crystallized the dilemma by stating that we can only give general recommendations for the user interface, which are based on general cultural characteristics of the human use of computers, but there is no warranty that the recommendations are appropriate in a specific case. A good example of situationality in HCI-research is the well-known set of three papers by Ledgard et al. (1980), Scapin (1981) and Landauer et al. (1983). The authors report studies that are concerned with the choice of command terms for text-editing operations. A casual reader seeking support for a particular design decision might justifiably claim the implications of the above papers somewhat confusing and contradictory. Ledgard et al. (1981) found out that 'familiar descriptive everyday words and well-formed English phrases' leads to better performance than a 'notational' dialogue. In an another study (Scapin 1981) it appeared that 'computer-oriented' terms were superior to 'task-oriented' terms for inexperienced subjects, while a third one (Landauer et al. 1983) implies that the choice of vocabulary did not produce substantial effects on the time taken to learn a system.

It has been pointed out (e.g. Jörgensen et al. 1983, Hammond et al. 1987) that the above results are only seemingly contradictory. For example, the studies used different measures of human performance and task conditions were dissimilar. From the point of view of a HCI-researcher these results are pieces of a very large behavioral jigsaw puzzle (Barnard 1983) and as Hammond et al. (1987) claim "the picture on the puzzle is unknown and many of the pieces are missing". The above studies show that our results are not only dependent on situational attributes, but also highly sensitive to our methods of inquiry. As Moher and Schneider (1982) report the behavioral scientists have long noted the effect of the experimental environment: subjects may be affected by the presence of others in the room, by the medium in which they communicate their responses, or even climatic conditions.

Due to the situational dependencies it is remarkably difficult to generalize the results beyond the specific conditions of the experiment and to adjust a right balance between internal and external validity in the design of an HCI-experiment. High internal validity tends to lead to limited generalizability, while on the other hand, the study by Whiteside et al. (1985) demonstrates the other side of the problem. Their experiment evaluated user performance with command, menu and iconic interfaces and the approach taken was "a holistic comparison of systems rather than decomposition into component system variables", because it could be assumed that "differences are due to an inextricably complex interaction of causes, so that decomposition is probably futile". As justified the previous type of claims may be, such research settings tend to arise problems, because it is nearly impossible to interpret the results reliably. In the case of Whiteside et al. (1985) were the performance differences (or the lack of them) due to help systems, the number of functions, the hardware characteristics or the interaction styles?

3. THE ROLE OF TECHNOLOGICAL PROGRESS IN HCI

Although a great number of user interface studies have been concerned with the assessment of various situational factors, their implications for user interface design is anything but clear. An exhaustive research program would require an enormous effort at the level situationality has been identified in contemporary HCI-research. This is painfully presented in Shneiderman's (1987b) article that offers seven issues and specific challenges (e. g. interaction styles and input techniques) for researchers and developers of human-computer interaction while he states that "each issue invites a lifetime of effort and a book-length survey". The range of situational factors that could be considered relevant in user interface research is wide and there is no agreed consensus what they should be. Our following discussion is based on a categorization that identifies

1) users
2) tasks and
3) technology

as the primary sources of situationality.

User needs and requirements are the starting point for the user interface research that aims to adapt the system to the user. Hansen's (1971) guideline 'Know the user' has been frequently cited over the past years. Current understanding of user requirements recognizes that users are a highly heterogenous group of people having different kinds of characteristics and needs. Due to this notable diversity of human capabilities and backgrounds the system users are an important source of situationality. The range of user characteristics considered in the user interface studies include, for example, psychological and demographical factors, such as cognitive style, education and experience. Those who are interested of a more detailed discussion on various user characteristics might find Rich's (1983) discussion on individual user models as a useful starting point.

The task dimension in a human-computer interaction situation is as diversified source of situationality as the user characteristics. The task dimensions considered vary from applications areas (e.g. decision support, computer-aided-instruction, process control, information retrieval) to more "elementary" type of tasks, such as data entry and choice selection tasks. The relevance of user and task dimensions of situationality have been covered relatively well in discussions on HCI, but the technological dimension has received much less attention, because the level of current technology is "given" from the point of view of a HCI-researcher. Situationality in the technological dimension is not so varied as in the other two dimensions, because due to the fast technological progress it is not worthwhile to do research in the old and less powerful hardware environments. Rather one should aim to work in hardware contexts that are on the verge of coming into everyday use.

If we were ready to take the challenge to conquer the mountain of situationality with an enormous research programme, we should be convinced that the reality under study is relatively stable. Otherwise, we are in the danger of being forced to start the process again and again from the beginning. The basic human characteristics change only marginally and the tasks the systems are used to are somewhat stable, but the technological dimension is in a state of constant flux. The pace of technological evolution does not show any signs of stagnation and we have well grounded reasons to believe that the future hardware environments will be capable to handle more and more difficult operations with the help of greater processing power, larger storage capacity, better physical characteristics of output and input devices etc.

It is trivial to state that new technological contexts cannot be studied before they have been constructed, but the statement shows that it is somewhat idealistic to look for a comprehensive situational framework. Such a framework requires the technological dimension to be effective, but as long as the technology evolves with the current pace, it will be at least partly defective. If the technological progress were to cease, we would still face the problem of situational complexity.

It is difficult to predict the forces that will be behind future developments, but by looking backwards we can gain understanding of the forces that have shaped the user interface design movement. Nielsen's (1988) classification of major user interface paradigms offers a good starting point for our discussion:

1. Generation: Batch (0-dimensional interactions: pass/fail)
2. Generation: Line/command-oriented (1-dimensional)
3. Generation: Full screen/form filling (2-dimensional)
4. Generation: Window/bitmapped interfaces (3-dimensional because of overlapping, multiple windows).

The overall quality of user interfaces has unquestionably improved in every shift from one paradigm generation to another. In fact, they are major milestones in the process of improving user interface quality and it is hard to find out any other process that has had more profound effect on the quality of user interfaces we are facing today. Are these paradigm sifts then results of controlled HCI-experiments? Is there any reported controlled experiment that has presented the clear advantages of these new user interface concepts before they have been taken into everyday use? The answer is no, there exists virtually no such experiment. Thus no controlled experiment could have resulted in these breakthroughs.

It is easy to see the role of technology as the driving force behind the different generations. The new ways of constructing user interfaces were not just possible before the necessary technological development. A teletype terminal behind slow communications lines is not the ideal environment to catalyze the invention of a direct manipulation interface. As Gaines (1986) has noticed the magnitude of quantitative change of underlying technology, although itself in continuous, leads to discontinuous qualitative changes as new possibilities become cost-effective.

The technological changes do not refute the old research results; they just bring up new questions that are more relevant under the new paradigm. For example, the question what character is the most appropriate to separate command language attributes is not a very central problem in the design of a direct manipulation interface. Due to the technological development it is difficult to find out research themes that have been highly relevant throughout the short history of HCI-research. The research interest seem to gradually cease after the first boom, although any "final" results have not been produced. The old problems are just not anymore relevant enough to be able to attract new research projects. For example, the effect of response times on user performance and satisfaction received attention in the end of 60's and in the 70's. Even though there is still plenty of unanswered questions in relation to response times they are not interesting problems in the current hardware context. Scientific achievements are results of cumulative research traditions. To be productive and advancing they need researchers' keen and steady interest on a subject over a longer period of time.

In relation to the present level of technology it is often claimed that good HCI (the current window/bitmapped display placed paradigm) had to wait for technology, but we should be highly aware that prevailing technological context is as constraining as it has been before. The concept of quality is relative to what you have witnessed before. The modern interfaces that use the desktop metaphor would look like extremely constrained in the eyes of a future user that is used to, for example, flying swiftly in a colorful three dimensional "artificial world" from one function to another and being able to issue speech commands. A statement by Walther & O'Neil (1974) gives us perspective: "As the initial fascination with conversational computing wore off, user reported experiencing feelings of intense frustration and of being 'manipulated' by a seemingly unyielding, rigid, intolerant dialogue partner....".

The above argumentation stresses the importance of underlying technology in the user interface design. It is likely to surface criticism, because the HCI-research community has traditionally emphasized the need to move from a technology based design to a user centered approach. Technology can be argued to be just a prerequisite medium for a cognitive interface that is the most important determinant in the user interface quality. Thus the research results dealing with the cognitive aspects can be claimed to be applicable even if the technology changes.

4. CONCLUSIONS

This paper challenges the deeply rooted belief of controlled experiments as a source of advances in HCI. This kind of introspective questioning has recently gained momentum within the HCI movement. For example, see the discussion that has been reported by Nielsen (1989) under the heading "Is HCI Research of Any use?". This study aims to contribute to the discussion by arguing that advances in HCI are inherently bound to the technological progress, and it is these technological developments that make it possible for the creative HCI designers to introduce novel ways to design the user interface. In the major breakthroughs the improved user interface quality is self-evident and it is grasped intuitively. Thus the breakthrough designs are not placed under thorough investigation and if they were, the new designs would be also subject to situational claims. The technological development bring up new research problems that are more relevant within new paradigms. Unfortunately, the results of controlled experiments have only a limited applicability, because the problems have extreme situational loadings.

The lack of concrete normative guidelines that could be recommended to practitioners with a clear conscience emphases the vital role of user interface designers. It is the task of a designer to create an interface that is capable to encounter the situational complexity. Behavioral experiments do not produce results that the could be directly applied by the designers. All experimental results and other suggested guidelines are dependent on their user for sensitive and intelligent interpretation. As a painting cannot be decomposed into single strokes without losing the essence of art, a good user interface is more than a simple sum of guidelines derived from experiments testing single a feature.

User interfaces of good quality are the result of a diligent, essentially creative, design process. User interface design is more like a creative art or engineering rather than an analytic science: "Design is art. All attempts to constrain it to be a top-down process are doomed. The finest user interface designs come to us as the gifts of creativity, forged from the long labor of craftsmanship. We can analyze and critique design in a highly structured fashion, but the aesthetics of an exiting interface defy formulaic generation" (Carey 1989). Also the technological development that has an essential, underlying function in the user interface design is in a similar way a product of creative design. "Actually the root word of technology, techne, originally meant "art". The ancient Greeks never separated art from manufacture in their minds, and so never developed separate words for them" (Pirsig 1974).

The role of experimental research is entirely different in natural sciences that study the "given" reality than in sciences that focus on human creations. One cannot experiment on something before it has been created; the emerging new and probably better designs make the problem area subject to mutability. It is the basic problem of sciences that study the man made artifacts of our environment. The mutable problem area gives strong arguments for a user interface research method that is sometimes called as "proof by design". The concept involves development work that is made in state of the art hardware environments and aims to produce novel user interface products (see e.g. Nielsen's (1988) description of the work that is being made in MIT Media Lab). This type of research is more centered towards the appreciation of potentials of new technologies than towards HCI theory building. Such research aims to new user interfaces where the sensation of a major breakthrough is so immediate that controlled experiments are not needed. As Negroponte (Nielsen 1988) has put it: "If one needs testing to prove one's point, then it is probably not worth doing".

Finally the reader should be warned of possible pitfalls of drawing some attempting conclusion on the basis of this paper. First, the problem of situationality complexity cannot be used as an argument to justify any kind of research method that aims to produce traditional type of results (normative statements about how the interface should be constructed). Whatever the research method may be, all obtained results are expressed in language. Thus all the results are subject to situational claims regardless of the method of inquiry. Second, the above conclusions should by no means be interpreted that conducted HCI-experiments have been useless. The experiments have certainly produced useful material that has been essential for our evolving understanding of the problem area. Third, the reader should be careful not to confuse experimental research with experimental development of specific products (such as usability tests). The latter kind of experiments are certainly useful tools in the way towards better designed products.

ACKNOWLEDGEMENTS

I would like to thank Pertti Järvinen, Markku Sääksjärvi, Ari Vepsäläinen, Timo Saarinen, Jukka Heikkilä and Bo Dahlblom for criticism, discussion and encouragement.

Financial support from the Doctoral Education Program of Information Technology in Finland and the Research Foundation of the OKOBANK group is gratefully acknowledged.

REFERENCES

Alty J., Mullin J. & Weir G. (1986). Survey of Dialogue Systems and Literature on Dialogue Design. Internal report no. AMU8601/01S, Alvey Man-Machine Interaction Unit, Heriot-Watt University / University of Strathclyde.

Barnard P. (1983). Experiments on Learning Interactive Dialogues: Problems and Prospects. In Elphick M. (ed.), Man-Machine Interaction, Proceedings of the Joint IBM/University of Newcastle Seminar.

Bodker S. (1987). Through the Interface - a Human Activity Approach to User Interface Design. Ph. D. dissertation, Aarhus University.

Carey T. (1989). Position Paper: The Basic HCI Course for software Engineers. SIGCHI Bulletin, 20(3), 14-15.

Cheriton D. (1976). Man-Machine Interface for Time Sharing systems. In Proceedings of the ACM National Conference, 362-380.

English W., Engelbart D. & Berman M. (1967). Display-Selection Techniques for Text Manipulation. IEEE Transaction on Human Factors in Electronics, 8(1), 5-15.

Gaines B. (1981). The Technology of Interaction-Dialogue Programming Rules. International Journal of Man-Machine Studies, 14(1), 133-150.

Gaines B. & Shaw M. (1983). Dialogue Engineering. In Sime & Coombs (eds.), Designing for Human Computer Communication.

Gaines B. (1986). From Timesharing to the Sixth Generation: the Development of Human-Computer Interaction. Part I. International Journal of Man-Machine Studies, 16, 1-27.

Gardiner M. & Christie B. (eds.) (1987). Applying Cognitive Psychology to User-interface Design, Whiley, New York.

Gould I. & Lewis C. (1985). Designing for Usability: Key Principles and What Designers Think. Communications of the ACM, 28, 300-311.

Hammond N., Gardiner M., Christie B. & Marshall C. (1987). The Role of Cognitive Psychology in User-Interface Design. In Gardiner M. & Christie B. (eds.), Applying Cognitive Psychology to User-interface Design, Whiley, New York, 13-53.

Hansen W. (1971). User Engineering Principles for Interactive Systems. AFIPS Conference Proceedings, 39, 523-532.

Jörgensen A., Barnard P., Hammond N. & Clark I. (1983). Naming Commands: An Analysis of Designers' Naming Behavior. In Green T., Payne S. & van der Veer G. (eds.), The Psychology of Computer Use, Academic Press, London, 69-88.

Landauer T., Galotti K. & Hartwell S. (1983). Natural Command Names and Initial Learning: A Study of Text-Editing Terms.

Communications of the ACM, 26(7), 495-503.

Ledgard H., Whiteside J. A., Singer A. & Seymour W. (1980). The Natural Language of Interactive Systems. Communications of the ACM, 23(10), 556-563.

Maguire M. (1982). An Evaluation of Published Recommendations on the Design of Man-Computer Dialogues, International Journal of Man-Machine Studies, 16(?), 237-261.

Marshall C., Nelson C. & Gardiner M. (1987). Design Guidelines. In Gardiner M. & Christie B. (eds.), Applying Cognitive Psychology to User-interface Design, Whiley, New York, 221-278.

Mitchell J. & Welty C. (1988). Experimentation in Computer Science: an Empirical View. International Journal of Man-Machine Studies, 29, 613-624.

Moher T. & Schneider G. (1982). Methodology and Experimental Research in Software Engineering. International Journal of Man-Machine Studies, 16, 65-87.

Nielsen J. (1988). CHI'88 Trip Report. SIGCHI Bulletin, 20(2), 58-66.

Nielsen J. (1989). HCI'88 Trip Report. SIGCHI Bulletin, 20(3), 90-93.

Pirsig R. (1974). Zen and the Art of Motorcycle Maintenance. Transworld Publishers, London.

Rich E. (1983). Users are Individuals: Individualizing User Models. International Journal of Man-Machine Studies, 18, 199-214.

Scapin D. (1981). Computer Commands in Restricted Natural Language: Some Aspects of Memory and Experience. Human Factors, 23(3), 365-375.

Shneiderman B. (1987a). Designing the User Interface: Strategies for Effective Human-Computer Interaction. Addison-Wesley, Reading, Massachusetts.

Shneiderman B. (1987b). Seven Plus or Minus Two: Central Issues in Human-Computer Interaction. In Carroll J. & Tanner P. (eds.), CHI+GI 1987 Conference Proceedings, ACM, New York, 343-349.

Walther G. & O'Neil H. (1974). On-line User-Computer Interface - The Effects of Interface Flexibility, Terminal Type, and Experience on Performance. Proceedings of National Computer Conference, vol. 43, 379-384.

THE SCENARIO-DOMINATED FUTURE WORKSHOP - CHANGE OR ADAPTION ?

In the Autumm of the 1988 the VISA-project[1] used a special version of the Future Workshop[2] as a Technology Assessment tool. The special version called the "scenario-dominated future workshop" was inspired from the technology assessment project called SELFI, which made workshops like these in the financial sector. (Andersen, Anne Marie and Hansen, Nils Ø, 1988)

Our aim was to discuss the role of Technology in future organizational structure in a Public Assistance Office.

In connection with planning these workshops we discussed the use of future workshops in Denmark till now. We studied the different aims behind using the workshops and the actual proces of a number of these, in order to avoid some of the problems we had experienced previosly, using the same type of workshop.

In this article I will present our consideration in the VISA-projectgroup regarding to the use of a number of scenario-dominated Future Workshops. After that I will discuss the results of these, in a change versus adaption perspective, for the participants. We operate with these perspectives in order to distinguesh between the use of the Future Workshop, in the system development and the technology assesment processes.

My intention is absolutely not to emphasize the excellence of the Future Workshops. I am more inclined to think, that the use of the tool often is problematic, because there is no reflection of why and how it is used.

I would like to have a debate about what our choise of tools and methods means for the participants, who are exposed to our selections, and what our choise means for the results, we get.

[1]VISA stands for "Technology Evaluation of Knowledgebased Systems for Management in Finance- and Administration Sectors". The Project is financed by Teknologirådet in Denmark. Members of the projectgroup were Oluf Danielsen, Birgitte Ravn Olesen and Karen Albertsen.

[2]Look at Annex A for a description of the traditional Future Workshop.

Future Workshops in Denmark

"The future workshop" was developed in the 50'ies in Western Germany as a reaction or a counterpart to the traditional growth-centered american research methods. The influence of these new ideas in Denmark did not arrive till much later.

The original aim was to create an atmosphere - a sense of being together enabling all the participants to take part -across intellectual, language or other professional barriers.

The starting point of the Workshop is the participants own experiences and demands. The aim; a communal development of plans of action, and the means; the unprejudiced communication. The leaders of the Workshop take part as initiaters -not as lectures.

Robert Jungk has been taking part in the development of the Future Workshops for many years. He states in his principal work, that the aim, is to support people in ways which lead to imaginations of how to cross borderline-changes in their everyday life. This Perspective-for-change is very essential to him. (Jungk and Müller, 1984)

Today the Future Workshops are used in numerous ways, some of which being far from the original intentions.

*Use of the Future Workshop on these occasions can be seen from different viewpoints. I have chosen to look at:

1. The different kinds of advantages for the participants taking part in the Future Workshop under different kinds of organizational frames

2. The different kinds of advantages for the system developer and researcher, in arranging Future Workshops under different kinds of organizational frames

Put in a scheme, the two viewpoints can be percieved in this way:

The organisational frame of the Future Workshop	Advantage for the participants	Advantage for the systemdeveloper	Advantage for the researcher
1. Possibilities of exceeding the social limits	*the participants are making all decisions by themselves *they have the possibility of obtaining an increased knowledge and power of their own worksituation *the theme is formulated in a way, that makes new solutions attractive	*the systemdeveloper works as a counter-expert *she obtains insight in the experiences and needs of the users, because she primarily listens and tries to help, when the users need her knowledge	*the researcher works on the conditions of the participants *she participates without having special demands to the process. She can later advantageoius refer to the results of the Workshop
2. Possibilities of changing the organisational and social limits	*the participants have a voice in the undertaking *they obtain increased knowledge within a beforehand well-defined field	*the systemdeveloper works as a consultant *she has defined the theme of the Workshop related to her own interests, which means, that she is sure of getting the knowledge, she needs in the developmentprocess	*the researcher arrange the Workshop, but is interested in the participants own experiences and interests *she participates and regulates the process in accordance with her own interests
3. Possibilities of adapting the organisational and social limits	*the participants have no influence *they obtain a break in the everyday routine, but the Workshop will not change the conditions for influence of the participants	*the systemdeveloper uses the Workshop as a tool to obtain contact with the users, as a part of a teaching-process or as a tool to legitimate implementation of allready finished systems	*the researcher arrange the Workshop without consideration for the participants *she get's the results, she wants, because the only role the participants have, is to be emperical material in the research-process

In the first columm, we have the organisational frame, closest to the "action for change-perspective". Within this frame the participants are making all decisions by themselves. They are responsible both, to the process and results of the workshop, and by making plans for action to follow-up the workshop.

The system developer, in this organisational frame, will act as an counter-expert. Finn Kensing charactarizes a "counter expert" as a person having starting point in a critical attitude to management ideas about computer applications, and by making dynamic agreements about the process with the employees. (Kensing, 1986)

She has no "hidden agendas" for the workshop.

The researcher can be seen as a "researcher-for-action". That means here a person, who participates in a way, so that her own ressources can be used, without trying to steer the interest of the participants.

As far as I know, very few of the system development and technology assesment projects, using the Future Workshop can be categorised in columm one, because most Future Workshops has been arranged in organisations, which beforehand have limited interest in the influence of the users. Closest to this columm is the Future Workshop used in ETB-planning and -development courses for union-activists.

Columm two contains situations, where some themes beforehand are defined as, open for discussion and influence, and others are not. That means, the participants are e.g. asked only for their opinion as to which system, that should be implemented, and not, if a given system should not be used in the organisation at all.

The system developer functions here as a consultant. She arranges the Future Workshop, because she wants to get knowledge about the users expectations and demands. She can use the Future Workshop as a sort of "negotiation-tool" in the developmentprocess, where she has defined the agenda.

The purpose of the researcher will be to make a process, she can observe, without loosing touch with her own research-field.

My opinion is, that most Future Workshops arranged in the system development and technology assesment projects is to be placed in columm two.

Columm three represents situations where the decisions really have been made. The Future Workshops prime function is to legitimate ready-made decisions.

The participants will have some amusing hours. Their experience can be, that they were asked, but they did not manage to use their chance to get influence.

The purpose of the system developer will only be to make contact or indirectly teach the participants. She will not take their opinions into account.

The researcher is just interested in empirical material related to her own research-process.

Unfortunately there are several examples of Future Workshops used as described in columm three. It is not necessaryly the irresponsibility of the system developers or researchers, that makes the Future Workshop, an adapting-process. Too little awareness of the existing organisational frames, within which the Workshop should be arranged and too naive confidence to the Future Workshop as an emancipating tool can place the Future Workshop in the situation described in columm three.

User-involvement without actual possibilities of getting influence, supports "adaption" and nog "change", in a more impenetrable way, than if there had not been any user-involvement at all.

To sum up: The Future Workshop does absolutely not in itself quarantee actual influence for the participants. It is a tool consonant with all other socio-techniques, and it can be used, and misused as these.

Alvesson, a swedish organisationresearcher, think's that the use of socio-technique methods in organizational development projects often will have a legitimizing effect on relations of effectivity-rationality and existing relation of power in the organisation. (Alvesson, 1983)

This statement is contrary to the very dominating attitude among many leaders of Future Workshops (Paaby, 1988 and Scott, 1988) who says, that the Future Workshop in the right conditions forms the basis of an open-minded communication. They have a supposition about the Future Workshop, as a method which (if practiced properly) creates room for the participant's own experiences and statements.

At the end of the article I will discuss where to place the scenario-dominated Future Workshops, we arranged at the VISA-project. Till then I'll have a brief discussion about some of the structures in the traditional Future Workshops.

It is obvious, that an aim in the "original" Future Workshop was to create "holes" or "punctures" in the dominant rationality, understood as dominating attitudes to what's to be seen as reasonable assumptions or demands. The participants must disregard existing power-structures, and instead believe in own experinces. But is that actually what happens in the Future Workshops ?

The Researchers in the HACKER-project[3] states the following after having arranged 4 Future Workshops concerning "Communication in the Health Care Sector":

> "... that the culture of "patient-treatment" and following from this reactive resolution processes is embedded as archaeological coats in the consciousness of all actors in the Health Care Sector - espescially among patients, the-next-of-kins and the politicians." (Anker Brink Lund, 1987)

Hereby I understand, that even if the mentioned groups express a deep scepticism and dissatisfaction with the existing Health Care system, then they themselves reproduce the pictures they react against.

It is not that easy to settle accounts with or revise the impressions of oneself, even if you experienced dissatisfaction with the realization of these.

Inside an organization everybody acts on the basis of certain impressions of for example which tasks they have, which room for personal creativity there is in the tasks and whom it is good to coorperate with. In other words there arises norms, routines and myths, which nobody in the organization in the daily life is aware of. Everybody accepts them as "the way things function". (Christensen, 1987) In the frame of the Future Workshop these norms, routines and myths are not put on the agenda. They are normally reproduced.

[3]"HACKER" is a Technology Assesment Project. It stands for Helsetjenestens Anvendelse af Computerassisterede Kommunikationsmidler (Use of Computer assisted communication in the Health Care Sector), and was made by Reseachers from Roskilde Universitetscenter.

In other words the Future Workshop has a tendency to leave the participants with their experiences. They are not impelled to make deeper reflections about the earlier mentioned limits and possibilities in the organisation. But these are necessary if not only superficial phenomenons, which are obvious in the dayly-life-experiences should be the result of the Workshop.

Naturally the experiences and prejudices of the participants are not independent of the social and organizational contexts in which they have emerged.

Changes occurs only if new knowledge and new experiences are presented. Otherwise the existing relations of power is manifested as being unchangeable.

Oskar Negt distinquishes between two types of learning-processes:

Deductive learning-processes, which takes the starting point in concepts, principals and lawfulnesses, where the participants acquires knowledge in a teacher-centered process.

Inductive learning-processes, which takes the starting point in the experiences of the participants. When the inductive learning-process is best it ends up with the participants own formulation of generalised ways of presenting the discussed problems.

Where the deductive learning process implies an adaption to existing knowledge and structures of power, does the inductive learning process make possibilities for a break upon these.

The Future Workshop is, in it's original form, an inductive learning-process. The problem of the inductive learning-process is, that the participants normally doesn't get a "tool" to belabour the commen experiences, which turns up in the process. They do not get a coherent and superior understanding of their problems. Without such an understanding it is difficult to change the discussed situation. The limits will be considered to be too strong. That means, that the Future Workshop can be almost as socialising to adaptiveness as the deductive learning process.

What is the Scenario-dominated Workshop?

Oskar Negt recomments in his work the need of making learning-processes, which imply contradictive examples. That means all learning-processes must involve both a empirical, experiencebased level and a generalised, theoretical level. (Negt, 1973)

In such contradictive processes the participants are socialised to embed the contradictions and make actions by themselves in relation to these. The aim of the Scenario-dominated Future Workshop is to establish such contradictive learningprocesses.

We see understanding and change as two parts of the same matter. This supposition means, that a change of existing structures of power and conditions of work claims a more fundamental understanding of these conditions. The participants in the Future Workshop process must cross the existing frames of understanding if new ideas about how things like tasks, division of labour, cooperation and management could be organized.

In the scenario-dominated Future Workshop we try to establish such contradictive acknowledgement and the mentioned "holes" in the existing frames of understanding. We do this by presenting the participants to some "pictures" or "Scenarios", which add new perspectives to their own experiences.

The scenario-dominated Future Workshop is in this way from the very beginning dependent on a higher degree of steering from the leaders of the Workshop, and the steering is in this case more obvious for alle parts in the Workshop.

The fact is, that most Workshopleaders does steer in the Workshop-process. In the last "deployment phase" new knowledge about f.ex. technical or juridical conditions are often demanded. The leaders must have the knowledge or they must help the participants to where they can obtain the knowledge. This information selection process can involve a high degree of steering. In the description of two Workshops arranged when a new etb-system should be implemented at Køge Hospital is e.g. said:

> "The groups could immediately before, and during the deployment phase, draw on help from experts. Because we had been able to foresee, that the participants would suggest etb-solutions, we had asked an etb-expert to be there. The groups consulted him to ask if their ideas could be technically realised. And they could ...."
> (Christensen, Ingrid and Jørgensen, Torben s. 74)

The participants are asked to relate some exact Scenarios of the future (which is drawn beforehand) to their own experiences. We had drawn 3 Scenarios of futural public assistance offices. The fokus was put on the structure of the administration and the technological demands, these structures would imply. The Scenarios was primarily drawn according to our general knowledge about development tendencies inside the public administration. We tried to make three general realistic Scenarios.

The participants were early in the Workshop confronted with these Scenarios. The consequence should be, that they were forced to imply their own experiences and wishes with certain aspects of the development process.

We would create room for profound reflections of the participants by (through the Scenarios) to confront them with the possibilities of the future. Reflection implies however also dialogue. That's the reason why we let the participants spend more time in smaller groups than it is common in the traditional Future Workshop, where headwords in a plenary is the main method op work.

Our intention using the Scenario-dominated Workshop was:

1. We wanted to establish a learning-process, that gave the participants a better understanding of own possibilities of action.

2. We wanted to collect material for our Technology Assesment Project in general.

Not only did we want to accumulate the criticism of the participants (rejection or minor adjustment) but we intended to support a creative process. The participants were hereby able to make demands or formulate wishes for their future work and position of technology in regard hereto.

We wanted to achieve insight in the employees ideas of an attitude to technology, when it is incorporated in a organizational structure, and not only presented as an isolated tool of technique.

The result was not intended to be generalized in a scientific sence. Contrary it was intended as the basis for estimating the attitude of different groups of interest to future public assistante offices.

Experiences with the Scenario-dominated Future Workshop

Before the Scenario-dominated Future Workshop took place, we had been involved in the evaluation of a prototype of a knowledgebased system placed in a newly established "Serviceshop"[4]. For that reason we were especially interested in that group. Through a serie of interviews with different groups of employees and managers we had achieved information inside the new organizational structure, culture, ways of communication and conflicts.

The municipal administration office had reasonly been through large structural differences and (espescially the managers of the organisation) was interested in superior discussion of future organizational structure and technological possibilities.

We had the responsibility not to press groups of adversary interest to a consensus agreement through a common Future Workshop. The Future Workshop is a method of developing ideas not a way solving conflicts.

On basis of our previous knowledge about the organisation we chosed to seperate the participants of the Workshops according to professions. We chosed groups that were most directly affected by the reasonably established changes. That meant agreements with the manager-group, the senior staff group, the employees of the Serviceshop plus a group of citizens. Each group were invited to participate in a Workshop of the length of one working day.

By establishing 4 different Workshops we gave each group of participants possibility to express themselves without the interference of the other groups.

---

[4]The Serviceshop is staffed by 8 female clerks coming from the former departments. Their job is to cope with first time applies and easily handed matters.

The idea was to demonstrate the relations of power and possible fields of conflicts in the Municipality by comparing the visions of the different Workshops to each other and to the existing strategy of planning. We herby wanted to point out whos interests was mainly taken care of in the existing plans for development. Our intention was to legitimate the existence of conflicts of interest and to point out how these conflicts so far had been handled in the planning process.

The 4 Workshops resulted in a number of experiences. Here is to be focused on two main problems:

1. The main problem for the participants:
   The Scenarios we had made were to general and too much out of touch with the experiences of the participants.

2. The main problem for us as researchers:
   The different development in the process of the Workshops made the comparison of results difficult.

About the Scenarios

Our Scenarios were charactarized by the fact that we were not clarified if we primarily wanted the participants to use their own experiences or if we were interested in more strategic contemplations.

The Scenarios did not imply the actual organization. We had not used our knowledge about specific situations in the public assistence office of this communicipality in developing the Scenarios.

If we had insisted strongly on the participants using our Scenarios we would have played the part of teachers and the Future Workshop would have become a deductive teaching process. This situation would have been like the situations described in columm three in the scheme. The participants would only have had a function relating to our ideas of what was usefull in our researchproject. The managers- and senior-staff-groups used the Scenarios, but did not exceed their own myths and norms. A condition to achieve that would have been a more limited field of operation. Therefore we left it to the participants to which degree they wanted to use our input.

We learned that the Scenarios must contain known as well as new aspects for the participants in order to achieve a brake of the frames of understanding.

In future we want to make Scenarios specificly aimed at the experiences of the participants and we want to focus on specific aspects. The aspects could be "division of labour" or "network contra Personal Computers in the office". This should make the Scenarios more familiar and interesting for the participants.

Creating such Scenarios implies thourough knowledge of the organization. It would be a good thing to have some of the future participants to comment on the Scenarios to be used in the Workshops.

The differences of the 4 Workshops

The participants of the 4 Workshops were very different. We had planned for some of the differences but more appeared in the process.

In his book on Research Interviews (Mishler, 1986) Elliot Mishler writes about problems of interviewing. One of his mainpoints being that the interviewer and the interviewed acts in a socio-culturel context. That means that the researcher must be very carefull when making and interpreting her questions because different interviewed persons will understand the same question in different ways according to their cultural contexts.

We did use the same Scenarios in all Workshops but the presentation and timing was different. We hereby tried to use our knowledge of the qualifications and experiences of the different participant groups. But we did not manage well enough. The somewhat general Scenarios favorised the senior-staff- and the manager-groups. Those were the groups that had the easiest access to the Scenarios and were able to present the most coherent ideas of the future.

The citizen-group consisted mainly of young unemployed people, that already knew each other. They were good at making phantasies of the future, but they refused to use our Scenarios as a basis. The Serviceshop-group had so many daily problems that they never managed to have an opinion about the Scenarios. In their behavior they insisted on having a tradional Future Workshop process.

The result was 4 different Workshop-documents, which could be compared to some aspects, but mainly stood for themselves.

The Results of the Workshops were not applyable with the strategical planning of the municipality. The documents of the Workshops simply were too incoherent and random.

The relation of power in the organization were not treated at all. Each group of participants stated their opinions, and we did not react to how the documents were used in the organization.

A more precise theme - maybe Scenarios which contained definite examples of the present strategy for new technique in the organization - could have discovered the conflicts of interests more easily.

Conclusions

Our idea was, that the scenario-dominated Future Workshop has two primarily advantages over the traditional Future Workshop:

1. The steering of the process is visible to everybody.

2. The contradictions embedded in the scenarios forces the participants to cross the level of spontanius experiences.

From my viewpoint it is obvious that we succeeded in having a visible steering of the workshops. In this connection I will regret, our steering was too powerfull, I regret that, it cannot be justified, by the fact, that it was obvious.

We did not succeed in establishing a learningprocess, that gave the participants a better understanding of their own possibilities of action, because there was too big a distance between our scenarios and the experiences of the participants.

Looking back upon the Scheme I presented at the beginning, I think the Future Workshops of the VISA-project must be put in columm two or three.

If I should argue in favour of placing the workshops in columm two, I would emphasize on the conditions in the organisation, where we were free to let the groups do, what they wanted to. We had no "hidden agendas". Our intention was to make a platform for change and/or uncovering the existing structures of power.

Furthermore we tried to adapt the Workshops to the different groups of participants in an attempt to let all groups have the same conditions for participation.

The conditions were good for creating a process, which made it possible for the participants to have a voice in the development, but as far as we did not succeed in this attempt, you may say, that our Workshops must be placed in columm three.

Our main fault was that we did not involve some of the participants in the process of arranging the workshop, and in drawing up the scenarios. This means, that we did not succeed in getting comparable results from the four Workshops, which spoiled attempts to compare the resulting demands of the Workshops, with the plan originating from Municipality. Our attempt to establish the same conditions for all participants did not work well enough. Our scenarios (and that goes for other academic performances) took place in a way, which favorised the body of power in the organization.

However the Workshops did have some positive effects:

- the Workshop process of the Serviceshop had an important function in changing individual frustrations into collective problems.

- the day after the Workshop the employees of the Serviceshop took a so far unused room into position as waiting room for clients. This deed was considered extremely provocative at the Workshop. A lot of energy was used ensuring each other that it could ve changed if anybody complained. The group experienced a positive response to the initiative. It was an important learning-process, that lead to a break with the former experiences "We Can't Do Anything!". The conclusion is, that the most concrete and visible changes appeared in the Workshop, which did not use the Scenarios.

- the Senior-staff members expressed after their Workshop satisfaction of having had time for a thourough and profitable discussion.

- all participants were pleased to see their ideas put into one report, which was to be handed to the politicians of the Municipality. The viewpoint of the 4 groups were published.

- and we have got some important experiences to use in our futural work.

A thorough description of the planning and results of the scenario-dominated Future Workshops can be read in VISA report Three:

References

Albertsen, Karen	Scenarieværksteder om Social- og Sundhedsforvaltningen - med ansatte og borgere fra Ringsted, VISA-projektet 1989
Andersen, Anne Marie and Hansen, Nils Ø	SELFI-RAPPORT nr. 7: Arbejds-forhold og selvbetjening i pengeinstitutter
Alvesson, Mats	Organisationsteori och teknokratisk medvetande, Natur och Kultur, 1983
Christensen, Ingrid og Jørgensen, Torben	Fremtidig kommunikation på et syge-hus, RASK i "Teknologiens muligheder - og menneskets", Industri og handelsstyrelsen, 1988
Christensen, Søren og Molin, Jan	Organisationskulturer, København, 1987
Jungk Robert og Müllert, Norbert	Håndbog i fremtidsværk-steder, Viborg, 1984

Kensing, Finn — Generation of Visions In Systems Development i "Systems Design for Human Development and Productivity" (edc: Docherty et al., North Holland, 1986

Lund, Anker Brink og Christensen, Jytte Møller — Kommunikation i sundhedsvæsenet 1 og 2, Aversi tryk hhv. 1986 og 1987

Mishler, Edgar — Research Interviewing - Context and Narrative, Harvard University Press, 1986

Negt, Oskar — Sociologisk fantasi og eksemplarisk indlæring, Roskilde Universitetscenter, 1973

Paaby, Nielsen og Aagaard — Fremtidsværksteder som foregrebet utopi i "Kontext 51", Viborg, 1988

Sørensen, Jette Scott (red) — Fremtidsværksteder i Danmark, Viborg, 1987

Annex A

Structure of the "traditional" Future Workshop

The Future Workshop itself consists of three phases. When practiced these must clearly be distinguished from each other.

1. **A criticism phase,** where the Participants in catch-words criticize the Theme on which the Workshop is founded. The Criticism is only based upon the Participants own Experiences.

2. **An imagination phase,** where the Participants on behalf of their criticism creates an Utopia, where everything is allowed. In the Course of this it is forbidden to squint at the world of everyday realities.

3. **A deployment phase** in which the Participants seek to convert the ideas from the imagination phase into actual Proposals, they themselves can handle and do something about in their everydag life.

Besides this there comes a **preparation** and a **picking up or action** phase.

INFORMATION SYSTEMS RESEARCH BETWEEN ARTS AND SCIENCE

Seppo Visala

Department of Information Processing Science

University of Oulu

90570 Oulu, Finland

Abstract: The article analyses information systems research (ISR) as an academic discipline. The focus is on possible and relevant research approaches. The paper tries to reconcile the conflict between two schools of metascience in ISR, positivism and hermeneutics, by pointing out their competence area within the field. For this purpose the well-known framework of Ives, Hamilton and Davis is reinterpreted in the light of different research approaches. The possibilities of the discipline to meet general scientific research requirements and, on the other hand, demands of creativity, are also briefly addressed.

1. INTRODUCTION

1.1 Structuring concepts

The title of the paper is deliberately ambiguous. You may read it as 'IS research between science and the humanities', and, on the other hand, it raises the question whether IS research as an academic activity may reach the status of science at all, or is it a branch of creative arts. The former question is the main issue of the paper. The latter serves here as a related frame to connect the subject into the theme of the seminar, but of course it is the crucial one as to the position of IS science in general.

Considering the possibilities of IS research in general, we must ask whether there are any means of theoretically mastering complex factors in the use and development of ISs. The possible theoretical constructs should aim at such general academic principles as *publicity, objectivity, criticality, testability, autocorrectiveness, autonomicity and advancement* (see e.g. Tuo83). These requirements do not explicate an unique scientific methodology. We may regard them as **ideal principles** of research, which realize themselves in any branch of science in a manner of their own. How far they may be maintained in ISS as a basically social science is a real challenge.

The prescriptive knowledge of ISD has been gathered into a vast amount of different development methodologies and models. Because of the diversity of methodologies and their generally rather poor ability to direct the development process, they are questioned all together, or more general 'multiview' models have been proposed (e.g. WH84). I will discuss the issue of theoretical constructs briefly in chapter 4. I will suggest the possibility of a macro model of information systems as a theoretical guide.

This paper focuses on the possible research approaches of IS (cf. Ehn88,5). I pursue these approaches through intertwined ontological and epistemological preunderstanding of the object of ISR. As such the paper is related to the lively paradigm debate in ISR, which is briefly reviewed in chapter 2. The general background of this study is the distinction of two schools of metascience, positivism and the hermeneutic school, which are also recognized in ISR. In this paper I will pursue a synthesis of these views based on the concept of preunderstanding the subject matter of IS research. My starting point is the common criticism of metaphysics by Heidegger and Wittgenstein as it is recognized by K-O Apel (Apel69). The core idea is their concept of a socially determined world interpretation through the practical relation of man to world.

The world is seen in different horizons, which also involve different possible research approaches. For a proper perception of our field of study, we must give up objectivism that seeks only causal explanations. I will propose a few tentative research approaches, which are used to reinterpret the well-known framework of ISR of Ives, Hamilton and Davis.

1.2 An IS research framework

As a rapidly growing new field of research ISS is heterogeneous, bringing about every now and then interesting new areas for exploration (see Coop88). To fix the target of the discussion I will use the widely adopted framework of Ives, Hamilton and Davis (IHD80), which presents the subject of ISR (see e.g. Coop88, Lyy87). As the subject is discussed in this paper, the compactness of the elements of the framework will cause some problems, and Ives' et al. treatment of the framework itself is subject to major criticism. The value of the framework as unifying the research area is not questioned.

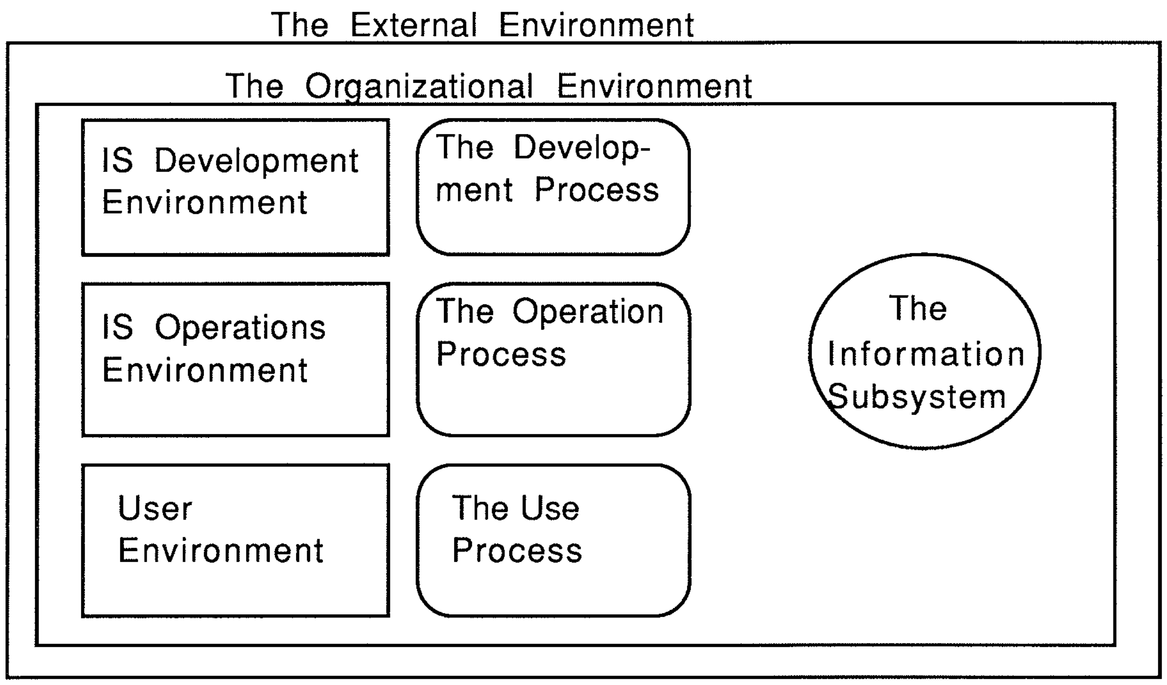

The External Environment
The Organizational Environment
IS Development Environment
The Develop-ment Process
IS Operations Environment
The Operation Process
The Information Subsystem
User Environment
The Use Process

Figure 1. A framework for ISR (IHV80)

I understand the IS concept as embedded in this scheme in a complex way, including social and technical aspects. The information subsystem (ISS) covers IS content and services. We should also remember that ISS models a universe of discourse (Iiv89). The framework will be used to indicate possible research approaches. The elements of the scheme are, however, too aggregated for all finesses to be opened.

2. ON THE PARADIGM DEBATE

There is no space in this paper to review thoroughly the by now rather comprehensive paradigm debate on the foundations of ISR. Besides there has been an extensive discussion on the very concept of paradigm since Kuhn introduced it in 1962 (Kuhn69, see e.g. Ver86). This philosophical argumentation has hardly ever been taken into account in IS field (see BaLa89). There are some recent papers (some of them still unpublished) that throw light on the debate from different aspects and give a comprehensive bibliography, such as HiKl89, BaLa89, Iiv89, and the proceedings of Manchester colloquium (MHFW85) are still of current interest. I will outline here only a few approaches to give a background to my own presentation.

2.1 Different approaches

The book of Burrell and Morgan, 'Sociological Paradigms and Organisational Analysis', has obtained a kind of paradigm handbook status in ISR debate (BuMo79). They name four major paradigms: 'functionalist', 'interpretative', 'radical humanist' and 'radical structuralist'. These arise from two dimensions, subjective-objective and social regulation-radical change. Hirsch-heim and Klein have recognized representatives of all these paradigms in IS as well (HiKl89). (They call the radical humanist paradigm neohumanism, i.e. subjective-radical change corner of the scheme.) Burrell and Morgan's framework has greatly facilitated synthetizing the fuzzy area. It may be, though, that it has exhausted its force to some extent (see Chua86 on the criticism of their paradigm concept). The subjective-objective dimension, after some modification, is also one focus of my paper. I will address the issue below.

The dimension of social regulation-radical change is certainly important in a broader sosiological context. One can question whether it is equally important in both radical structur-alism and neohumanism in ISD context. DEMOS and UTOPIA projects are examples of the class conflict view. Yet the radical change through social emancipation is a more general concept. I do not regard neohumanism as really opposed to the functionalist view in the regulation-radical change dimension (this same remark applies to Lyytinen's thesis (Lyy86), which is the first major contribution introducing Habermas' critical theory in ISR). I will overlook here the regulation-radical change dimension as far as it concerns macro level social structures.

Banville and Landry (BaLa89) criticize on good grounds the Kuhnian monolithic concept of paradigm, which assumes that at the mature stage of any science only one model of research dominates. Kuhn's concept has originated mainly from the history of natural sciences. With paradigm criteria taken from the ideal state of physics, the social sciences are doomed to stay on the preparadigmatic level. As an interesting alternative they trace out ISR paradigms along Whitley's societal theory of intellectual fields. That theory grasps fields as observable social constructs, not so much as a priori preconditions of research. The social and institutional aspect of research communities is important and has been addressed also partly by Klein and Lyytinen (KlLy84) and Nurminen (Nur86). My viewpoint here will in any case be that of an a priori ontological and epistemological basis of research, but as I hope to show, this basis itself is of social origin.

Iivari (Iiv89) has written a paradigmatic comparative analysis of contemporary schools of ISD. He has selected seven major schools, which can be traced back to existing textbooks. As to the ontological and epistemological foundations, Iivari has classified all those schools as belonging mainly to the objectivist side. Applying general paradigmatic concepts his paper analyses existing schools of IS development. My focus is on the philosophical background of these concepts as a basis of IS research.

2.2 Revision of Burrell and Morgan

Let us look more closely at Burrell and Morgan's definition of the subjective-objective dimension. These opposite views are aggregated standpoints to four sets of assumptions related to ontology, epistemology, human nature and methodology. In these issues they give as ideal extreme positions realism and nominalism (idealism), positivism and anti-positivism, determinism and voluntarism, and nomothetic and ideographic methods respectively. The former alternatives make up the objectivist approach and the latter the subjectivist one. Although it may be questioned whether these positions are too aggregated considering the various existing lines of research, one must admit that this kind of confrontation is a commonly accepted view in philosophy of science (see e.g. Wrig71).

However, we should not accept their setting of 'subjectivism' as an opposite to objectivism. This is mainly due to their philosophically rather superficial definitions of ontological attitudes. Burrell and Morgan define nominalism as follows:"*The nominalist position revolves around the assumption that the social world external to individual cognition is made up of nothing more than names, concepts and labels which are used to structure reality*" (BuMo79,4). Realism, again, "*... postulates that social world external to individual cognition is a real world made up of hard, tangible and relatively immutable structures.-- For the realist, the social world has an existence which is as hard and concrete as the natural world*" (ibid).

The characterizations are problematic and do not leave much choice. It is hard to deny the existence of social structures in some sense, but it is inapproriate to equate that to the existence of natural objects. Their definitions are misleading in that they assume some kind of uniform concept of existence which can unequivocally be attached to everything that we regard as real. This 'existence' probably comes close to how we believe to see natural things without any qualifiers given by our own practical relation to those things. The individual cognition is the only one not falling into the same category of the objectively existing world. It is this objectivistic idea of being, which is behind both the given ontological views, and that is why 'subjectivism' is given as the only alternative of objectivism.

The first thing we ought to do is to clarify the meaning of 'being' (*Sinn von Sein*), which Heidegger has set as the basic task of philosophy. We cannot speak of the being of man, natural objects and social structures in the same sense. We have not the right to define realism with criteria which distinguish the constitution of the world only as concrete and sensuous, and, on the other hand, as practical and meaningful. I will address the issue more deeply below, and only hint at the solution here to name the alternatives more adequately.

We can use the term **objectivism** here in the sense that the objectivist objectifies the being of everything in the same sense as the being of the natural objects without any qualities arising from the practical relation that human beings have to them. The term subjectivism is unsuitable, since giving sense to things is not an individual property but due to the social nature of man. I prefer here the term **humanism**. The standpoint of the humanist is that man sees the world in the first place through meanings, which have originated from his social and practical relation to world, from his own way of being, existence (Heid27).

2.3 Two schools of metascience and ISR

Those two ontological positions are more or less the basis to two opposite views of metascience. When we especially want to point out their epistemological and methodological conceptions we might call them positivism and hermeneutics (Wri71). The terms, 'positivism' at least, have become rather trite expressions, and are used as labels without any further consideration. I use them according to von Wright's chacterization (Hirschheim gives rather similar characterizations, Hir84). The core assumptions of positivism are 1) methodological monism, 2) exact natural sciences as the ideal of research and 3) seeking causal explanations as a true purpose of science. Hermeneutics challenges all these principles. The concept of understanding as opposed to causal explanations has been recognized as its distinctive feature.

As to the conception of human nature, the advocates of these schools may adopt determinism as well as voluntarism. But in the case of positivism, voluntarism effectively gives up the entire possibility of understanding human behaviour scientifically (cf. Wittgenstein's *Tractatus* and moral action of human beings). In the case of hermeneutics voluntarism fits to the concept of understanding, but determinism is also possible to some extent, if cultural tradition is regarded as an authority (Gadamer). Determinism in the sense of positivism leads to treating human beings as natural objects.

These schools can be also recognized in IS research, positivism as an implicit assumption of the main stream of research in the field, hermeneutics as a criticism of positivism and a manifesto, but by now also in concrete research projects (HiKl89).

3. PHILOSOPHICAL SYNTHESIS

Opening basic philosophical questions once more is problematic, for to be of any interest at all the philosophical point of view should be able to problematize 'the natural view'. I feel, however, that the paradigm debate has not yet reached the ultimate core of the issue, that is, the ontologically and epistemologically intertwined preunderstanding of the subject matter of ISR. I myself can surely not claim to have solved the problem here, but only point to some ideas.

3.1 Ontological issue

Taking ontological questions genuinely into consideration is a kind of a philosophical choice itself. Classical ontological categories seem old-fashioned and the analytical school has considered the issue only 'from the logical point of view'. Wittgenstein's later period does problematize it anew, but has not settled the question finally and reflectively. Heidegger's fundamental ontology has been regarded as a rather peculiar sort of thinking. Karl-Otto Apel has seen a close relationship in the thinking of these two philosophers (Apel69). It is not possible here to enter Apel's paper or the writings of those two philosophers in detail, but only to raise the main point as far as it concerns our topic.

Indeed, there is a relationship in how late Wittgenstein criticizes logical atomism (metaphysics of *Tractatus*) from his philosophy of language, and introduces with pragmatic semantics the world interpretation through the concept of the form of life (Witt52), and in how Heidegger builds the interpretation of the meaning of being with a phenomenological analysis of the being of man (Heid27). They both disprove traditional metaphysics, which seeks the constitution of the world just looking at it out there, transcending human cognition, forgetting how man's practical stand to the world constitutes the world. As Heidegger puts it: things are not just simply there, but they are ready to hand (*zuhanden*), to be used for something. Only after their ready-to-hand-ness has been stripped away, can one see them as natural objects (*vorhanden*).[1]

To avoid too lengthy speculations, let me to give a rather naive example. It is easy to become convinced that we cannot define the concept of table by any set of external qualities, nor can we point out any unequivocal set of tables as a meaning of the 'table'. We can regard as a table any object we **use as** a table. To quote Apel:"*Not an animal or a pure spirit, but only a human*

[1] I find Winograd and Flores' interpretation of these concepts slightly confusing (WiFl86), as they use Heidegger's fundamental ontology to their own purposes of designing technical equipment with the limited idea of 'breakdown'.

being, who is in an understanding relation to his existence as a possibility, can 'let something be' 'a table' or 'a chair', i.e. can equip the world with meanings".[2]

In a form of life the world unfolds as meaningful. The world interpretation given by a form of life is not a product of an individual but of a community. Persons own actions are given meaning by the community that he or she belongs to. I read a book as a book is usually read, like everybody does. To Heidegger *das Man*, a neutral everyman, is a basic existential quality of man. "*Jeder is der Andere, und Keiner er selbst*" (Heid27,128). In more realistic terms, the world is socially constructed (BeLu66).

To both Heidegger and Wittgenstein, language is a central concept. Using Wittgenstein's concepts we could say that the world interpretation is articulated in a language game. The concept of language game has broadened to a metaphor, which has been used to cover the whole social life of man (e.g. Winch58). Though one cannot deny the importance of language as an essential feature of man, there is, however, a danger in over-emphasizing social actions only as language-like rule following. We forget thus other more substantial, material conditions of social life. Apel considers these two great thinkers somewhat ahistorical, but sees great potential in their ideas through materialistic interpretation.

To continue this line of thought, I may summarize it as follows (cf. Ehn88). The world interpretation is articulated in language, but we encounter the world by means of our practical actions. Man is also part of nature. Through socially organized praxis man renews the necessary conditions of his own existence under some constraints, which are to some extent unavoidable natural laws and partly historical and evolving social and economic circumstances. Economic 'laws' do not determine individual acts, but they are important constraints of acts within a social **horizon** that delineates the meaning of acts. A human being faces choices, and some other horizons can overcome 'economic facts'. Organizations and other social structures are forms of life among others. They are horizons of people's actions, and, on the other hand, people maintain these structures through their actions, which must satisfy certain conditions set to the organization by external society. These structures outlast individual people, but they surely do not last for ever. Organizations are part of a socio-economic system which regulates their scope of action. They are not harmonious communities with no conflicts at all.

The description is rather common one. The point here is that when searching for theoretical means to deal with organizations and people's acts, we ought to see them through different horizons that always give our theoretical endeavour some preunderstanding of the subject of the research. Futhermore, information systems model a part of reality, a universe of discourse

[2] My own translation from the Finnish version of Apel's paper hopefully suffices.

which ought to be understood as socially constructed, too. The form of life of people **constitutes**[3] the universe of discourse.

3.2 Epistemological issue

The epistemological question is, how do we capture a better hold of the already preunderstood world. Preunderstanding is the starting point. It is an a priori condition of knowledge in ourselves, which is socially constructed and ever evolving. Even though we cannot find any eternal truth, we have, however, always some basic assumptions that we take as granted. There is no proof that there are invariable natural laws, but we take it as granted. Nothing proves that man has free will, but at least some of us do believe so. Violating these assumptions even in the name of science in one way or another would shake the very foundation of all rational behaviour. For instance, if one sees people acting reasonably and meaningfully, and tries to nullify the intentionality of action, he or she violates their own understanding of human beings.

The Concepts 'explanation' and 'understanding' imply a great deal of preunderstanding that directs the scientist in his research. The objectivist tries to manage with explanations alone, the humanist advocates understanding, perhaps at the cost of explanation. I will briefly discuss these concepts according to von Wright's ideas (Wri71).

Hempel's *covering law model* is a generally accepted presentation of explanation. A phenomenon is explained by known preconditions and a set of general laws, which together imply the phenomenon. The laws are not logical but contingent causal laws. von Wright defines causality as a necessary non-logical relation. The necessity means that the laws do not just indicate certain repeatedly observed chain or coincidence of events. The law can be accepted only after experiments that cover all conceivable situations in which the hypothetic chain of events is either produced or prevented as expected. The law has also contrafactual force, i.e. we can conditionally claim that the effect would be realized,if we only brought the causes about (which we don't). So the laws are tested instrumentally (cf. Hab63).

Can we explain acts causally? von Wright gives good grounds against that. The essential feature of an act is its purpose or aim. His main argument against the causal explanation of an intentionally understood act (i.e. that the purpose is not a cause of the act) is the following. We cannot assure ourselves of the nature of an act without knowing its purpose, and, on the other hand, we cannot be certain about the intention of the actor, if we do not see the act in some context. So we cannot recognize the cause and effect as logically independent phenomena, which is required by the definition of causality.(see Wri71, 93-)

[3] As to the terms 'horizon' and 'constitution', I have learned them from Husserl (Hus13), though I use them here in a more mundane sense.

The behaviourist might remark that the preceding argumentation cannot get rid of 'understanding' in favour of explanation, for 'understanding' is taken as an implicit prerequisite of an act, and one should only pay attention to what is 'really happening'. But here we encounter a different preunderstanding of acts, and the dispute is unsolvable. In any case, we are faced with understanding and hermeneutics, and as far as acts are concerned, we are faced with double hermeneutics.

When 'explaining' acts von Wright proposes replacing the covering law model by *practical syllogism*:

> A wants p;
>
> A knows that without doing a, he cannot obtain p;
>
> So A sets about doing a.

We could call that a teleological explanation. This model can be applied to describing individual acts, but it is not adequate for explaining collective acts (like demonstrations) or social structures (Wri71,132-). There is room for such theoretical constructs that model the action space within these structures. This is the opportunity of cybernetics (see Nis85). These models do not capture behaviour into deterministic laws, but describe the conditions of the development of structures (organizations in economic system, human society in eco-system) and different possible effects of acts chosen by people. They can use their knowledge of the possibilities to plan their action (Nis84, Nis85).

These models do not totally explain human behaviour. Why for instance do people do business? Here we collide with the cultural, social nature of man, and with those fundamental existential features through which we always see ourselves (in the Heideggerian sense), as being-in-the-world with others. There is always an end to all explanations: this is how things are. It is a challenge for hermeneutics to seek ever more profound horizons of understanding.

When we develop information systems, or computer based support systems in general, we sooner or later meet the requirements of logic and physical conditions (things *vorhanden*). To build systems that work as intended we must rely on formal methods. Even if system builders do not carry out physical research, they must trust the operation of the technical equipment.

The moral of the lecture is that there is not just one proper means to grasp the rich reality, but that we must approach it from different horizons. In the case of informations systems all above mentioned approaches are relevant, some perhaps just marginally. Beyond all of them there is (more or less) free discussion between people participating in the development process, speculation and critique of all biased conceptions, creation of new conceptual insight, construction of tools and just collection of facts in a more or less ambiguous horizon. These approaches are called critical discourse, phenomenological seeing of essences, constructive method and statistics.

3.3 Research approaches and the framework

Let us revert to the subject matter of information systems research as presented in Figure 1. Ives, Hamilton and Davis do not open the framework in respect of different horizons of preunderstanding, for they share the positivistic assumption of methodological monism. They see it only as a framework for vaguely defined causal explanations. I have argued that the subject matter ought to be approached in different horizons by appropriate methods. As a matter of fact there may be many methods within each context of preunderstanding. Methods are not necessarily bound to one approach.

The framework of Fig. 1 is not ontologically sufficiently differentiated, so that we could associate unique approaches to all elements in it. For instance 'development environment' includes people, organization and methods. Some processes of development and evolution of information systems are complex in themselves; the learning process can be mentioned as an important example. In any case, through the ontological and epistemological extension of this framework we can open a richer view into the research than the narrow causal projection of Ives et al.

Table 2 displays the epistemological dimension over the framework of Figure 1. The most suitable research approaches of the elements themselves are found on the diagonal. The intersections of the elements show the relevant approaches to their interactions. On this level of abstraction the table only gives the main ideas and requires further deepening. The table is constructed on a rather intuitive basis.[4] The reader might notice an obvious relationship of the approaches to Habermas' knowledge interests (technical, practical and emansipatory, Hab63).

As epistemological approaches I have chosen the statistical collection of facts(S), formal methods (M), causal models (P), cybernetics (C), teleological explanations (T), hermeneutics (H) and aggregated critical discourse and philosophical conceptual analysis (phenomenology) (F). There is another aggregated approach under the title 'learning process'(*), for it is an important aspect in researching some features of information systems but is hard to bring under any one research approach.

Statistics has been chosen as an approach of its own, for it is related to the fuzzy horizon of observation which is open to different interpretations. Statistics is definetely not a method of

[4] Juhani Iivari supported the idea and made valuable comments on the table. I took them into account in finishing it off. I am also grateful to him for seeing the communicational value of the table more clearly than myself and advising me to devote more effort on developing my first sketch of the table.

proving causalities in the sense given above. Cybernetic models are separated from causal ones, for these are not approriate for self-steering systems. The term 'cybernetics' here represents a whole set of formal models that include also indeterministic elements. Teleological explanations cover intentional people's actions and interactions between them and different groups. These three terms should be seen as candidates for referring to the relevant approaches of respective horizons. As to the choice of other approaches, I refer to the previous section. The approaches and horizons are gathered together in Table 1.

APPROACH	CHARACTERIZATION OF HORIZON
Conceptual analysis (F)	Phenomenological reduction of the object,and discourse about concepts
Hermeneutics (H)	World of meanings, culture, and form of life
Teleological explanations (T)	Purposeful acts of individ-uals and interest groups, the political 'game'
Statistics (S)	The fuzzy horizon of un-classified observations
Cybernetics (C)	The structures and systems that can be captured in non-deterministic models
Causal models (P)	The instrumentally controlable world of technology
Formal methods (M)	Software, data models and description languages

Table 1. **Research approaches vs. horizons**

We may state the general features of the Table 2 as follows. Environments of different levels effect dialectically each other. The ontologically 'lowest' level (subject to technical knowledge interest in Habermas' classification) effects upper levels causally as a resource. Upper levels make up a social, cultural and economic environment to lower levels and give a hermeneutical interpretation scheme.

Let us look at the details. The essential features of the elements of the framework are on the diagonal, and they direct mainly decisions concerning the connections of elements. Most intersections are not relevant. The essences and relations interest us only as far as our subject matter, information systems and their development, is concerned. The table is triangular. Mutual effects of elements are in one intersection, though it may come out in more thorough

Components:									
1 External Environment	FHCS								
2 Organization Environment	HTC	FHCT							
3 Development Environment	HP	HT	FHM						
4 Development Process	H?	T	HT*	HT*					
5 User Environment	HTS	HT	-	H*	HPS				
6 Use Process	-	C	-	-	TP*	TP			
7 Operations Environment	P	-	CT	-	-	-	PM		
8 Operation Process	-	-	-	-	-	TP	P	PM	
9 Information Subsystem	H	TC	-	T	HT	-	-	-	HM
Components:	1	2	3	4	5	6	7	8	9

Table 2. **Research approaches of IS science**

For a legend see Table 1; in addition, * refers to aggregated approach into learning process (dialectics ?)

analysis that effects are not similar in both directions. (This form of presentation is more readable and safer at this stage, as I have learned from commentators of my paper.)

External environment is the cultural and socio-economic system where the organization operates. It gives the general interpretation background to everything else (symbol H in intersection 1/1) and it dictates also the material and social conditions for the continuation of that form of life (C:1/1). As a complex phenomenon it is always subject to conceptual analysis and must often be approached by statistical methods (FS:1/1). Some limits have been set to the organization environment by the external environment. It forms a form of life of its own (FH:2/2). This is largely directed by organization structure, strategic lines of the business idea, collaborative work and interaction of interest groups (CT:2/2).

Development environment includes people, development organization, methodologies and techniques. These make up the language game of development (H:3/3). Methodologies include conceptual models and formal methods as means of rational planning and design (FM:3/3). In the development process there is information production and interpretation, decision making, control and learning (HT*:4/4).

User environment consists of people, who should be understood (H:5/5). Also their general physical and psychical qualities are important (PS:5/5). The use process is determined by acts of users(T:6/6), their properties and the quality of the system (P:6/6). Operations environment is mastered instrumentally (PM:7/7). The operation process is based on technology and controlled by programs (PM:8/8). The information subsystem/content is a formal model of the universe of discourse (HM:9/9, cf. Iiv89b).

Next I shall look over the relevant mutual relations of the elements by columns. The external environment delimits organizational culture and the domain of ergodicity (HC:2/1). The organizational environment reacts to the external environment by strategic (T:2/1) and material (C:2/1) activities. The external environment also affects the development and user environments, since people act in many ways outside the organization, too (H:3/1,4/1;HT:5/1), and gives technological resourses among other things (P:3/1,7/1). Activities of organization in social environment constitute the universe of discourse that information system models (H:9/1). Due to the complex relationships that the organization has to its environment the statistical approach may be appropriate in more cases than is shown in the table.

Organization is the basic form of life of the development environment (H:3/2). Organizational decisions make up the development group (T:3/2) and direct the process (T:4/2). The information production and decision making of the development process have effects on the organizational environment as well (T:4/2). The interaction of different interest groups intermediates many effects of the IS on the organization (T:5/2). The functions of the organization determine the lines of the use process (C:6/2). Thus some of the effects are due to conscious decisions, others to structures and culture. The information subsystem is a functional resource of an organization and the information produced affects the decisions (TC:9/2).

The development environment has institutionalized certain rules (H:4/3) and operation models (T:4/3) concerning the development process. The learning dynamics of the process react on development environment (developers and methods) (*:4/3). The development environment may delimit the operations environment by economic conditions or for practical purposes (CT:7/3).

Users should be taken into account understandingly (H:5/4) in developing the IS. Users learn a lot, if they participate in the design process (*:5/4). The information production and decision making of the development process have effects on the definition of the information system (T:9/4).

IS imposes the rules of a language game on users. On the other hand, their life world and insight influence the information system content. Their actions maintain and modify rules of the information system (HT:9/5). The use process teaches users and consumes their resourses (P*:6/5). Their decisions and characteristics direct the use process (TP:6/5), and through that the operation process (TP:8/6). IS operations environment and operation process are together a real machine (P:8/7).

4. RESEARCH VS. PRACTICE DISCUSSION

4.1 The macro model and creativity

Whatever the research approach is, the research should satisfy the scientific principles mentioned in the introduction as rigorously as possible. The issue is not very problematic in the case of formal approaches, since they have obtained an accepted status as academic disciplines,i.e. as far as their objectivity etc. is concerned. Their applicability to human behaviour may be questioned, and I have delineated their power in accordance with the hermeneutical view. In the case of those approaches that are seen relevant to the study of the human action, the principles of objectivity, testability and autocorrectiveness are generally regarded as problematic. A thorough examination of these questions is beyond the scope of this paper.

Yet, we may briefly address the issue whether the research results can be presented in a theoretical form that supports the IS development in practice. Any theoretical construct would satisfy the scientific principles at least to the extent that it explicates in a concise manner the current knowledge on ISD and thus submits it to public examination.

Iivari has divided IS research into *conceptual*, *descriptive* and *prescriptive* levels (Iiv83,5). The descriptive level produces explanations and information that supports understanding of IS use and development, and the prescriptive level produces development methods and tools and recommendations for their practical use. Both these types of information are used reflectively in the development process. Considering the great diversity of possible research approaches we may doubt the possibility of a general theory of ISD. On the conceptual level, however, we may seek for **a macro model** which directs research and development. The macro model carries pointers to this (descriptive and prescriptive) knowledge and raises new research motifs.

I see the possibility of a macro model in common cultural characteristics of organizations in industrial societies, rather than in some stable qualities of human beings or in a general development methodology. We can grasp the potential common components of information systems within a period of time. The components change, of course, but not quicker than the organizational cultures in which we study information systems. A possible way to arrange these components is to submit them to predetermined abstraction levels (Iiv89b). These levels also outline the development process as a decision process. Yet they leave room for the reflective and creative building of the systems.

The issue of the ISD model is as old as the art of ISD. It has been one of main topics in earlier Scandinavian seminars in the late 70s and at the beginning of 80s. I have two reasons to raise the question on the model once more. For one thing, I agree with those who deny the possibility of a general development process model, except for the decision points that arise from the information system models at the various abstraction levels. For another, I believe in the possibility of gathering the potential components of information systems under a macro model (or competing models), which is adequate over a period of time.

The main argument for these models arises from the necessity to explicate the background assumptions behind every development process, and thus to submit them to public scientific scrutiny. The models may be imperfect, and they must be put to test every now and then. Yet, they always beat the unarranged reports of a thousand projects, since these do not contribute to the theoretical capital of research. There are also reasons to believe that a model maintained by the scientific community is more competent than any set of tacit practical rules of thumb that every practitioner must follow.

The concept of the macro model raises the question of creativity vs. theoretical mastering of ISD. No model can solve the unique practical problems of a development process. It can only point out problems that are to be expected and to the knowledge which may help to solve the problems. As we saw, this knowledge is of a very diverse nature. The knowledge carries with it some 'invariances', some common features of information systems. Next I will hint at a way of searching these invariances, which is an alternative to the traditional positivistic view, and opens up potential to creativity, too.

4.2 Understanding and creativity

We could see IS as a support to human activity, which is to be understood in a context of a form of life. A community can expand its form of life, it can even learn a new form of life when it is implementing an IS. There are more effective and more efficient ways of doing things. There are always some general and some contingential constraints, but once an IS has proved to work in one situation, it is at least possible for it to work in another, if the external conditions are sufficiently similar and people are ready to change their way of doing things. Even if people are not willing to alter their customs, the possibility remains. They may change their minds. Goldkuhl and Lyytinen have pointed to the concept of a life form mainly as a basis of a world interpretation and a language game (see GoLy82,Lyy86,260). My emphasis is here in a common human action.

To start a new model of activities is not just a matter of decision, of course; it also demands a lot of work to learn the new system. The point here is that IS is based on some general human *form of action,* a form of life. The descriptive research gives us information on (possibly changing) preconditions and constraints of actions. These we must take as a starting point on which we create new systems. The prescriptive research gives us conceptual and augmentative tools. A macro model gathers this knowledge in accordance with our conception of information systems at a given time. This we may use to support our creative endeavour.

I use the familiar metaphor of 'form of life' here to challenge the traditional causalities-seeking objectivism as the only possible means of finding any 'invariances' in ISR. Seeing IS use and development as a kind of general human activity is ontologically a better ground for capturing invariances into IS theory.(Of course, 'invariance' is not the proper term, since activities change as time goes on.) We should not look mainly for causal explanations but for general cultural basis for understanding human actions restricted by necessary logical and causal constraints. ISR cannot so much predict as direct the ISD.

ACKNOWLEDGEMENT

The paper is based on my master thesis, which I wrote under guidance of Ph.D. Juhani Iivari. I am much obliged to him for his keen comments on this paper as well. I am also grateful to Prof. Pentti Kerola and other colleagues of my department for their comments and inspiration, as well as to Finnish Doctoral Programme of ISS for the opportunity to hear many famous researchers, such as Heinz K. Klein, to mention the most influential one as far as this paper is concerned. My thanks are also due to commentators in IRIS, Skagen, especially Lucy Suchman. I have also here an opportunity to thank Tauno Tönning Foundation for financial support.

REFERENCES

Apel69	Karl-Otto Apel: Wittgenstein und Heidegger, *Philosophises Jahrbuch 75,* München 1967, pp. 56 - 94.
BaLa89	Claude Banville, Maurice Landry: Can the Field of MIS be Disciplined?, *Communications of the ACM*, January 1989, Vol. 32, No. 1, pp. 48 - 60.
BeLu66	Peter L. Berger, Thomas Luckmann: *The Social Construction of Reality, a treatise in the sociology of knowledge*, (1966), Penguin Books, Middlesex 1981.
BuMo79	Gibson Burrel, Gareth Morgan: *Sociological Paradigms and Organisational Analysis*, (1979), Heinemann, London 1982.
Chua86	Chua, W.F.:Radical developments in accounting thought, *The Accounting Re view,* VOl. LXI, No. 5, 1986
Coop88	Randolp B. Cooper: Review of Management Information Systems Research: A Management Support Emphasis, *Information Processing & Management,* Vol. 24, No. 1, pp. 73 - 102.
GoLy82	Göran Goldkuhl, Kalle Lyytinen: A Disposition for an Information Analysis Methodology Based on Speech-Act Theory (1982), in *Scan83,* pp. 201 - 238.
Ehn88	Pelle Ehn: *Work-Oriented Design of Computer Artifacts*, Arbetslivscentrum, Stockholm 1988.
Hab63	Jürgen Habermas: *Erkenntnis und Intresse,* (1963), Suhrkamp, Frankfurt am Main 1981.
Heid27	Martin Heidegger: *Sein und Zeit*, (1927), Max Niemeyer Verlag, Tübingen 1979.
Hir84	R. A. Hirschheim: Information Systems Epistemology: An Historical Perspective (1984), in *MHFW85*, pp. 13 - 35.
HiKl89	Rudi Hirschheim, Heinz K. Klein: *Four Paradigms of Information Systems Development* (1989, still unpublished).
Hus13	Edmund Husserl: *Ideen zu einer reinen Phänomenologie und phänomenologischen Philosophie*, Allgemeine Einführung in die reine Phänomenologie, (1913), Max Niemeyer Verlag, Tübingen 1980.
IHD80	Blake Ives, Scott Hamilton, Gordon B. Davis: A Framework for Research in Computer-Based Management Information Systems, *Management Science*, Vol 26, No. 9, September 1980, pp. 910 - 934.
Iiv83	Juhani Iivari: Contributions to the Theoretical Foundations of Systemeering Research and the PIOCO Model, (diss.) *Acta Universitatis Ouluensis,* A 150, Oulu 1983.
Iiv89	Juhani Iivari: *Contemporary Schools of IS Development: A Paradigmatic analysis,* (1989, still unpublished).
Iiv89b	Juhani Iivari: Levels of abstractions as a conceptual framework for an information system, to appear in Falkenberg, E.D. and Lindgren, P. (eds.): IFIP WG8.1 Working Conf. on Information Systems Concepts: An In-depth Analysis, Namur, Belgium 1989

KlLy84	Heinz K. Klein, Kalle Lyytinen: The Poverty of Scientism in Information Systems (1984), in *MHFW85,* pp. 131 - 161.
Kuhn69	Thomas S. Kuhn: *The Structure of Scientific Revolution,* Second Edition, The University of Chicago Press, Chicago 1970.
Lyy86	Kalle Lyytinen: Information Systems Development as Social Action: Framework and Critical Implications, (diss.), *Jyväskylä Studies in Computer Science, economics and statistics 8*, University of Jyväskylä, Jyväskylä 1986.
Lyy87	Kalle Lyytinen: Different Perspectives on Information Systems: Problems and Solutions, *ACM Computing Surveys*, Vol. 19, No. 1, March 1987, pp. 5 - 43.
MHFW85	Enid Mumford, Rudi Hirschheim, Guy Fitzgerald, Trevor Wood-Harper(eds): *Research Methods in Informations Systems, Proc. of the IFIP WG 8.2 Coll.,* September 1984, Manchester, North-Holland, Amsterdam 1985.
Nis84	Hans-Erik Nissen: Acquiring Knowledge of Information Systems - Research in a Methodological Quagmire (1984), in *MHFW85,* pp. 39 - 51.
Nis85	Hans-Erik Nissen: A Theor-Ethical Basis for Studies of Information Systems Use and Development (1985) in *Scan85*, pp. 1 - 22.
Nur86	Markku I. Nurminen: *Three Perspectives to Information Systems,* Studentlitteratur (forthcoming, in Finnish: WSOY, Juva 1986).
Scan82	Göran Goldkuhl, Clas-Olof Kall (eds): *Report of the Fifth Scandinavian Research Seminar on Systemeering*, Gothenburg 1982.
Scan85	Monika Lassen, Lars Mathiassen (eds):*Report of the Eighth Scandinavian Research Seminar on Systemeering*, Aarhus 1985.
Tuo83	Raimo Tuomela: *Science, action, reality, philosophical foundations of scientific world view.* Gaudeamus, Jyväskylä 1983 (in Finnish).
Ver86	Veli Verronen: The Growth of Knowledge, an Inquiry into the Kuhnian Theory, (diss.), *Publications of the Department of Philosophy, University of Jyväskylä,* No 35, Jyväskylä 1986.
WH84	Trevor Wood-Harper: Research Methods in Information Systems: Using Action Research, in *MHFW85*, pp. 169 - 191.
WiFl86	Terry Winograd, Fernando Flores: *Understanding Computers and Cognition, A New Foundation for Design,* Ablex Publishing Corporation, Norwood, New Jersey 1986.
Win58	Peter Winch: *The Idea of Social Science and its Relation to Philosophy,* (1958), Routledge & Kegan Paul, London 1977.
Witt21	Ludwig Wittgenstein: *Tractatus Logico-Philosophicus*, Kegan Paul 1923.
Witt52	Ludwig Wittgenstein: *Philosophical Investigations,* (1952), Basil Blackwell, Oxford 1978.
Wri71	Georg Henrik von Wright: *Explanation and Understanding*, Routledge & Kegan Paul, London 1971.

The Decentralization of Oslo: Theory and Practice

Astrid Marie N. Reksnes
Leikny Øgrim
Kristin Aardal
University of Oslo, Dpt. of Informatics,
POBox 1080, Blindern,
N-0316 Oslo 3,
Norway.

1. INTRODUCTION

Oslo City is decentralized into 25 Urban Neighbourhoods. The objective of the decentralization has been to reduce expenses in the health and social services, and at the same time improve the quality of the services. A large project organization was established to carry out the reorganization and the development of the computer systems DATSOS (DATa Social OSlo) and PRHIS (PRimary Health service Information System). The computer system will have up to 20 000 users and about 500 000 people will be affected as clients and patients.

The system development project had a very tight time limit. More and more external consultants have been employed, with approximately a hundred people participating. The level of ambition has changed. Officially chosen methods have not been used. There have been constant delays and budget deficits. There has been problems with the management, and a lot of critical attention has arrived from the mass media.

Our original intention was to investigate the relation between organization development and system development. More precisely: was the aim of the decentralization a guiding line for the system development project?[1] From this outset we have restricted the paper to the discussion of two important aspects, keeping the goals of democratization in mind:[2]

1) Shortness of time. the political authorities regarded the connection between the organizational decentralization and the fulfillment of the computerization crucial. The dead line for a running computer system was fixed to the date of the decentralization, giving a project period of 1 1/2 year. We discuss this strong linking in Section 3.

[1]Reksnes will in her master thesis examine how the democratization aspect is taken care of in the project, and conclude with a suggestion of how to ensure real influence from the future users.

[2]We use the concept "democratization" in a common, intuitive way, as the possibilities for the future users to influence the development and use of the computer systems. For further discussion of the concept, see for instance Bjerknes et al.(87). Some aspects of the relation between decentralization and democratization are discussed in Øgrim(87) and Øgrim(88).

2) Project management is essential, especially in large system development projects such as this one. In Section 4 this is discussed in relation to the democratization aspect.

We discuss the central development strategy, and propose an alternative approach — prototyping in local health and social offices. This is discussed in section 5.

METHOD OF RESEARCH

We have not participated in the system development project. As a part of the work with two master theses[3] we have had interviews with central participants in the system development project, and a study of project documents. The theses are not fully carried out, and the information is far from complete.

The municipality of Oslo is an extremely bureaucratic organization, where apparently no one dares give any information to anyone without asking permission from the superior. Hence our knowledge is incomplete, and perhaps even wrong.

An important experience is that researchers in bureaucratic organizations need contracts and alliances with the management responsible of the subject of research.[4]

2. A BRIEF DESCRIPTION

In this section we want to give you a feeling of what the system development project is all about. Firstly, we give an outline of the application area, the Urban Neighbourhoods. Secondly, we describe of the project organization. Thirdly, we give an overview of the computer system, the section ending with a chronological review of the system development process.

2.1. An Urban Neighborhood in Oslo

On the 1st of July 1988, the administration of health and social services in Oslo was divided into 25 Urban Neighbourhoods, as localized administrative units. 20 000 employees were transferred from central to the Urban Neighborhood administrations, together with one third of the central budget. More functions will probably be decentralized in the following years, the idea being that the Urban Neighbourhoods will gradually increase their local autonomy and evolve into local municipalities.

The administration and formal tasks belonging to the Urban Neighbourhoods are built upon the same legislation. Nevertheless, they vary in many respects, such as geographical size and the composition of the population. To give an impression of the Urban Neighbourhoods, we will describe one of them, Søndre Nordstrand.[5]

Today, about twenty thousand inhabitants are living in this Urban Neighborhood,

[3]Reksnes (forthcoming) and Aardal(forthcoming). Leikny Øgrim is their tutor

[4]This experience is also mentioned in Bjerknes(89)

[5]Søndre Nordstrand will be further described in Aardal(forthcoming)

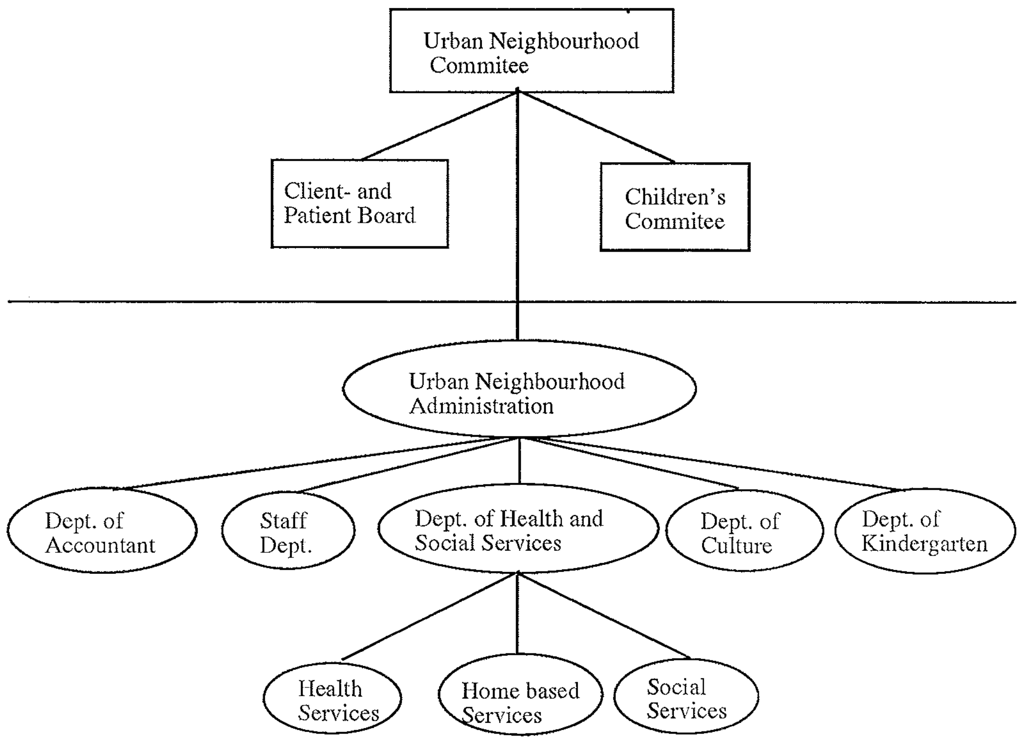


Figure 1. The Urban Neighbourhood Administration

and the population is increasing very fast. The plan is to reach the total number of forty (maybe even fifty) thousand people. The rapid expansion started in 1976 (with only a few thousand inhabitants). The population is fairly young.

The decentralization led to a new organizational unit, the Urban Neighborhood Administration. Before the decentralization, the operative health and social services were administered from central professional departments, which are now closed down. The Urban Neighborhood Administration is separated into two parts, one political (figure 1, rectangular boxes above the line) and one administrative (figure 1, circular boxes below the line). The political part consists of the Urban Neighborhood Committee with underlying committees and boards.

The administration is responsible for carrying out the resolutions made by the politicians. As shown in figure 1, the administration of Søndre Nordstrand consists of several departments, each of them with specific responsibility domains. The total amount of employees is about 700.

Søndre Nordstrand has two health centers administered by medical doctors. Public

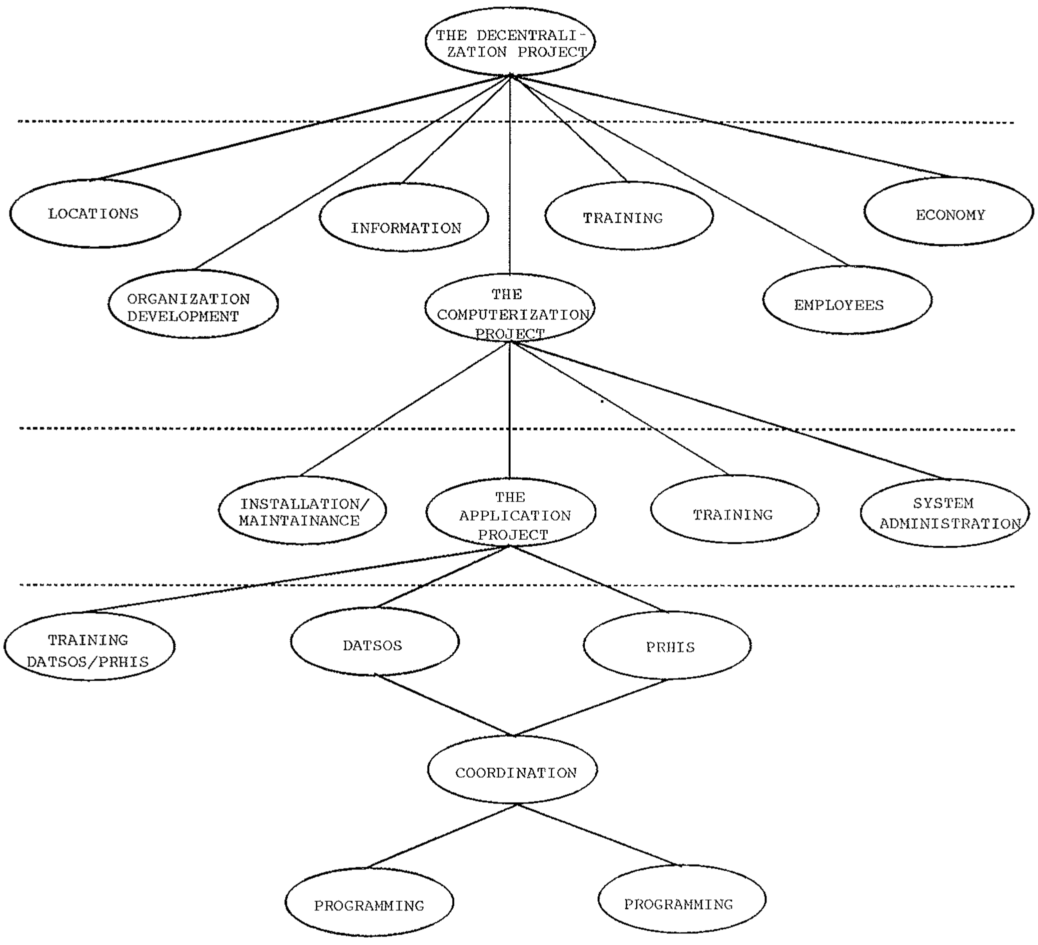


Fig 2: Project organization

health nurses govern the four maternal and child health centers. Besides, there are some private health workers with special agreements with the local government.

The three welfare offices give advice and guidance to the inhabitants, and distribute the economical social service.

All home based services are administered from the Urban Neighborhood. The home help is given to elderly and disabled persons to avoid sending them to institutions. Every inhabitant is, in case of acute sickness, entitled to home help. Homes for the elderly are also administered by this department.

2.2. Project Organization

Figure 2 shows the computerization project as a part of the huge organization of the decentralization project. As you see, there were several projects, similar to the computerization project. The computerization project had its own project leader, and an executive committee, with participants from some central departments and services and the trade unions involved.

The computerization project was organized in several sub-projects. Our study concentrates on the application project.

The application project was responsible for the development of the two computer systems DATSOS and PRHIS, and thus organized two separate groups, responsible for their several system. Both groups have been organized with a project leader and a project group. The leaders have changed during the project, and so have the group members.

Hence, when discussing project organization and project management, we thus talk about four different project levels: 1) decentralization, 2) computerization, 3) application and 4) DATSOS and PRHIS.

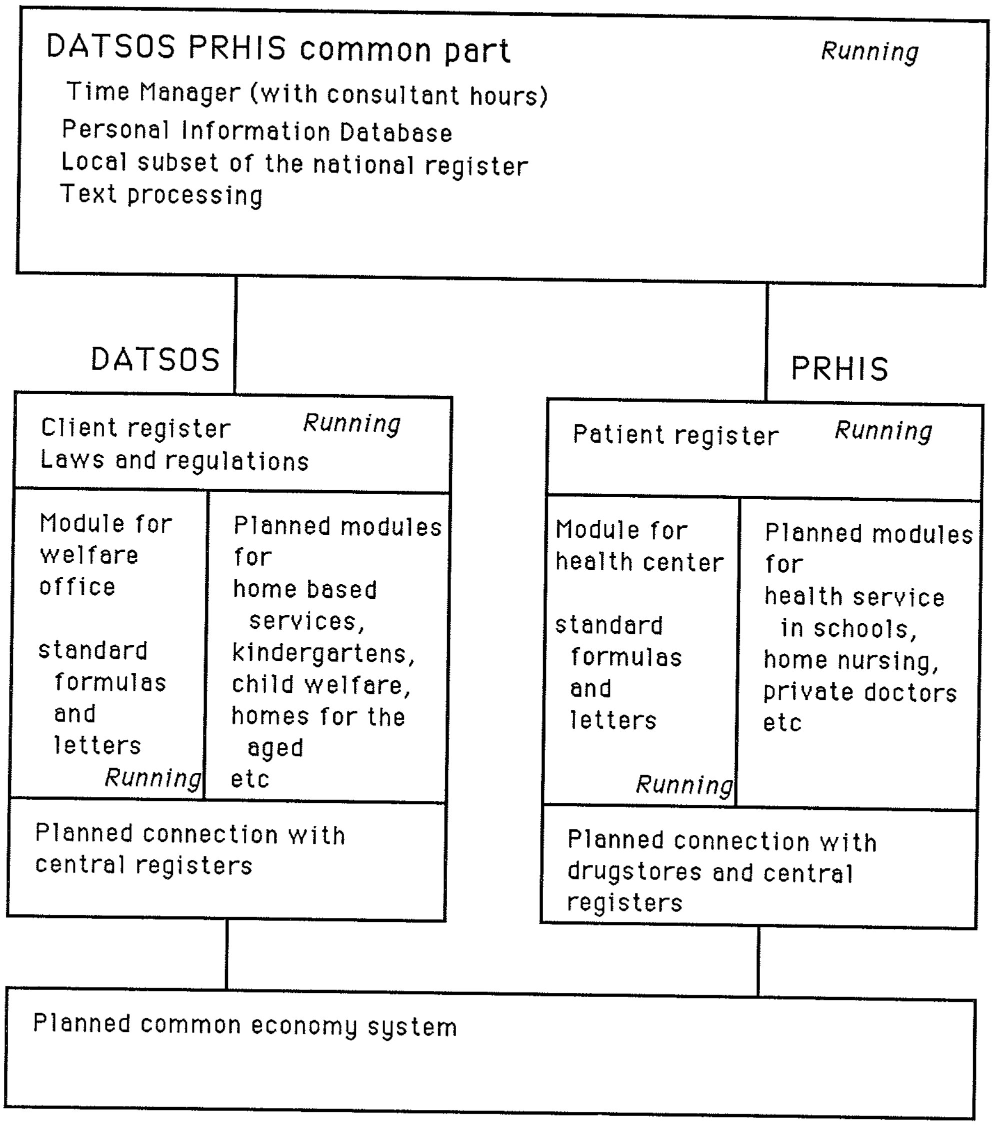
2.3. DATSOS and PRHIS

The principle of equal treatment of clients and patients is central in the health and social services. The quality of the services must not be dependent on where a person is situated. A national examination of living conditions showed that the largest social differences are found within Oslo. To prevent a reinforcement due to the decentralization, it was considered important that the Urban Neighbourhoods had common computer systems which gave comparable data to central political and professional authorities. On this basis, it was decided to develop computer systems which covered the needs of all the Urban Neighbourhoods, the systems adjustable to local needs.

Figure 3 shows the main functions and the status of the different modules of the two separate computer systems: DATSOS and PRHIS.

Norwegian laws do not allow an employee in the health services access to the same data as a welfare officer. DATSOS and PRHIS will be implemented on one machine, thus, restrictions of data access is handled by logical barriers in the system. If the laws are changed, the data access belonging to a profession can be changed easily.

Connection with a number of central registers are planned, but not yet effectuated.


DATSOS PRHIS common part
Running
Time Manager (with consultant hours)
Personal Information Database
Local subset of the national register
Text processing
DATSOS
PRHIS
Client register
Running
Laws and regulations
Patient register
Running
Module for welfare office
standard formulas and letters
Running
Planned modules for home based services, kindergartens, child welfare, homes for the aged etc
Module for health center
standard formulas and letters
Running
Planned modules for health service in schools, home nursing, private doctors etc
Planned connection with central registers
Planned connection with drugstores and central registers
Planned common economy system

The common part of DATSOS and PHRIS is almost complete, and is operative in one health center and one welfare office. The modules for health center and welfare office are data modeled, and the next step is programming. The intention has been to design several other modules, but these are only vaguely defined.

2.4. Chronological Review

January 1985: Local requirement study at one health center.

Spring 1985: Preliminary investigation on computer based systems for the municipal health and social services established.

Spring 1986: Preliminary project. Rough data models by user representatives.

September 1986: The project groups DATSOS and PRHIS. The date of the decentralization, January 1st 1988.

Autumn 1986: User Requirement Specification and data modeling. PRHIS on health centers, DATSOS on welfare offices.

January 1987: User Requirement Specification for health center finished.

Winter/early spring 1987: Coordination of the data models for the common part. System construction of the health center module and the common part. PRHIS making data models and User Requirements Specification for maternal and child health centers.

Autumn 1987: New date for decentralization, July 1st 1988.

October 1987: Hardware and application tools decided.

Spring 1988: Preparing of the common part for testing in the user organization. The system deadline, January 1st 1989.

May 1988: the test version installed in one health center and one welfare office. The health center decided to keep the version for further use. The social center went back to the old manual routines.

July 1st 1988: THE DECENTRALIZATION. 8 Urban Neighbourhoods without localities.

Autumn 1988: Data modeling and programming. The municipal budget for 1989 did not include further development of DATSOS and PRHIS.

January 1989: DATSOS and PRHIS closed down.

May 1989: The computerization project leader resigned.

July 1989: The project closed down. New consultant firms employed to analyze the future of the project. Supposed to be finished in november 1989.

3. THE LINKING

As mentioned earlier, the dead line of a running computer system was linked to the date of the decentralization. This led to extreme shortness of time. To try to keep within the time limit, more and more consultants were employed, but the extensive use of consultants was very expensive, and it was one of the reasons for the rampant budget deficits. Another reason for the employment of consultants was to make use of their special competence on different areas, but the differences in development philosophies and working habits led to new time-consuming discussions and, thus, further delays for every new consultant firm involved. If the total work is divided among more people, the amount of coordinating work is increased relative to the total work.[6]

[6]Weinberg(71)

The tight time limit led to an enormous labour. It turned out to be impossible to carry out all the intended activities within the time estimated. The project leader spent the first year on hardware negotiations, while the project groups concentrated on user requirement specifications and data modeling of the health center and welfare office parts of the systems.

Information to, and communication with, the Urban Neighbourhoods were held at an absolute minimum. The employees in the Urban Neighbourhoods knew very little about what was going on in the project, and were thereby not able to influence the process.

Intentionally, the systems were to be tested in four Urban Neighbourhoods. Due to the budget deficits, and the severe delays, the system tests were restricted to one health center and one welfare office.

At the health center there were activities to adjust the system to the organization. After three weeks, the local health center decided to keep the preliminary version for further use. The welfare office was not satisfied, maybe because necessary interfaces to several central registers were not implemented, and perhaps they needed local adjustment and development.

There was no parallel organization development in the other health centers, nor in the welfare offices, even if the officially chosen system development method emphazises parallel organization and system development.

The necessary relation between organization and system development is obvious.[7] So is the need for computer support in the health and social services. The decentralization created new needs for control and assurance of equal treatment of the clients and patients in the different Urban Neighbourhoods.

Our research has shown that there has been no parallel organization and system development. It is therefore tempting to ask whether the coordination in time has been used as a pretense to get finances from the municipal politicians, both regarding system development and hardware supply.

Let us for a moment imagine this linking had not happened. We would then probably have experienced discussions and questions in the municipal parliament. The politicians would have asked about goals, needs and financial costs. The probability of a cost/benefit analyses would have increased, and so would the possibilities for more careful planning and wide spread requirement surveys, including parallel organization development. In this way the conditions for influence from the employees in the Urban Neighbourhoods could have increased.[8]

[7]See for instance Järvinen(87), Kaasbøll(88), Bjerknes et al.(87), Avison and Fitzgerald(88)

[8]It seems to us that the systems DATSOS§PRHIS may be useful, but, due to the fact that they are incomplete and, thus, not implemented in the Urban Neighbourhoods, we ate not able to give any profound statements on how they would be in use.

4. MANAGEMENT

This is a large project. The project organization is complicated and involves different interests. In such a project, strong management is of great importance.[9]

The project leader's professional background was from the management of an institution for alcoholics, and from work in the Town Hall. In addition he had participated in the preliminary investigation on computer systems to the Health and Social Services in Oslo. Apart from this, he had no knowledge of computers, system development or project management.

The project leaders at the other levels[10] also lacked experience from project management.

There has been an obvious contradiction between the time limit and the quality ambitions. This contradiction has emerged as a conflict between the leaders of the different levels in the project. The project leader insisted on the time limit, while the sub-project leaders insisted on the original ambitions. The project participants wanted to do a proper job whatever time spent.

The project management was not able to handle this conflict. The weak management combined with the hierarchical project organization made internal communication difficult. Communication and mutual understanding are important factors in a democratic process. To communicate, there must be possibilities for a direct dialogue between employees and the management. If the structure of the organization is hierarchical, this direct contact may not exist.[11] An interesting aspect is the contradiction between the organizational ideal expressed by the decentralization, and the structure of the computerization project.[12]

Even though time limits were exceeded, reports to political authorities said that there were only small delays. The vast amount of philosophies and views upon system development, might have been used to create new ideas and to enforce a better process. The lack of leadership in the computerization project seems to have led to a situation where the consultants stressed their own views upon system development, with consequences to communication and democracy.

The user representatives' suggestions for changes were in most cases implemented, resulting in further delays. Nobody took the responsibility of restricting the design process.

Attempts have been made to improve the management. A project planning tool, called milestone planning, was bought. The philosophy is roughly like this: 1) Discussion and description of goals or objectives, 2) Description of activities necessary to

[9]Andersen et al(86), Yourdon(82), Avison and Fitzgerald(88)

[10]see figure 2

[11]Gustavsen(86)

[12]This discussion will be developed in Reksnes(forthcoming). She will in her master thesis examine how the democratization aspect is taken care of in the project, and conclude with suggestions of how to ensure real influence from the future users.

reach these goals, 3) Estimation of time needed for the activities, made by those who will carry them out, and 4) Estimation of the total time needed. When, as in this case, 1) is unclear and 4) is fixed, it is obvious that the technique is not used (even if the project group has spent a week in London attending a course of the technique, and even if the schemes are formally used).

We do not want to criticize the retired project leader. Instead, we find it more relevant to question the employment process.

Project leaders do not need to be system developers, nor do they have to know much about computers. But there has to be someone with good knowledge of organizing projects, to make decisions and to be able to coordinate the work. A project can not be managed effectively by persons who do not take part in the execution of the project.[13] It is possible that problems could have been reduced if people at lower levels of the organization had been allowed to take more decisions, or, at least, had been heard when they reported difficulties in the development.

A strong management will not bring democracy about automatically, but without the ability of conducting, being able to ensure progress in a project, democracy will suffer. The project ends up in endless discussions among the participants, and few applicable results.

5. DEVELOPMENT STRATEGY

The system development project has been carried out by a centralized project organization, as described in section 2.2. The project designed a common data model followed by a common computer system to be used in 25 different Urban Neighbourhoods, each with several health centers and welfare offices. Representatives from two Urban Neighbourhoods have participated in the project groups. Local testing and organizational development activities have been reduced to a minimum.

This strategy emphasizes the equalities between the local organizations. The wish for equal treatment of patients and clients and the necessity of following the health and social welfare regulations have been important in the system development process. These aspects have been well taken care of.

On the other hand, there exist many differences. The inhabitants in the Urban Neighbourhoods vary for instance in age, earnings, education and nationality. Housing standards, air pollution and amount of car traffic differs. The health centers and welfare offices have local variations in working organization. This is especially important in the welfare offices, that — due to heavy work load — almost continuously go through organizational changes. The differences have not been taken care of in the project. We find that the local democracy has suffered at the cost of the need for equal treatment.

[13]Andersen et al(86)

Only a few of the 25 Urban Neighbourhoods were represented in the central project group during the whole project period. From a democracy point of view, this is worth criticizing. We are not convinced that the central development was a wrong decision. May be it was necessary considering the short time available, the small budgets and the legitimate needs for equal treatment and central control.

Still we question that no other development strategy has ever been discussed, neither in the municipal authorities, nor at any level of the development project.

We propose an alternative strategy, based upon local prototyping in the Urban Neighbourhoods. We suggest this strategy to be used in further development of DATSOS and PRHIS and in future development projects supposed to serve the decentralization of the municipality.

Pape and Thoresen[14] report an experience with a strategy of this kind. Their paper deals with the problems of designing a system which will suit the different needs of several municipalities. Furthermore, they discuss how change processes can be organized in order to develop knowledge necessary for the municipality to manage technological and organizational change on their own, i.e. without undue dependence on the vendor or on centralized competence. This problem focusing seems to be very relevant to the present task, a well functioning computer system in the Urban Neighbourhoods.

We propose a two step development, the first a successive development of pilot versions in one or a few Urban Neighbourhoods. The second, a municipal survey to identify the spectrum of variations relevant to the design and use of a common data system for all the different Urban Neighbourhoods in Oslo.

The development of the pilot version starts with an analysis, to give the basis needed both for the design of the first prototype and the organizational implementation. Aardal[15] will, in her master thesis, use Soft Systems Methodology[16] to conduct an analysis of the needs for computerization in the Urban Neighborhood of Søndre Nordstrand.[17] This kind of local analysis will focus on the problematic situation of the decentralization and computerization, instead of pinpointing the equalities between the Urban Neighbourhoods and between the health centers and welfare offices. Soft Systems Methodology uses systems models in an organized learning system to improve the process by which "what obviously ought to be done" in an organization emerges.[18] We think the democratization idea will have better conditions under these circumstances.

The analysis should be combined with local prototyping. The most important feature in prototyping, as with Soft Systems Methodology, is the mutually learning

[14]Pape and Thoresen(87)
[15]Aardal(forthcoming)
[16]Checkland(81)
[17]see section 2.1
[18]Checkland(85)

and communication aspect between users and developers.[19] Prototyping may be used as a means of communication between the users and the system developers. Rapid prototyping of alternative functions and screens can stimulate fantasy and imagination[20]among the users.

As stated earlier, in line with our democratic outset, we think of prototyping as a good means of making the pilot version. In her paper on experimental prototyping, Wegge,[21] defines experimentation as the endeavor to test the realization of an idea by actively changing a given situation, changes in a situation where something which does not yet exist is to be constructed for the first time, or where elements are selected to form a new constellation from something which already exists. To our point of view, this should be most suitable in the decentralization project of Oslo, where there are both new things to come (the new organizational units the Urban Neighbourhoods), and a rearranging of existing structures (the old health and social services) and processes (report lines and communication between politicians, municipal officers and employees).

Even though exploratory prototyping is closest to the original meaning of the term "prototyping", it can only be recommended if there are tools available which keep to a minimum the effort required in constructing the prototype, and if the expected life time of the system is long enough or the quality requirements are sufficiently high to warrant the extra investment.[22] At least, the last two holds for our example, as we expect the computer system investment in the municipality to be of a size that large, that it *has* to be used over a long period of time. The economical situation too, points in that direction. And, due to the fact that the intentions of the computer systems are, amongst others, to take care of the treatment of social welfare clients and health service patients, one should expect the requirements of quality to be pretty high.

When the prototype is running in one Urban Neighborhood, it is time to bring in other participants. The result of an evaluation of the pilot version, will be the input to the next prototype and any organizational changes, as part of a next iteration. During this development period, a representative from a second Urban Neighbourhood should sit in on project meetings, to learn about the system and the organizational implementation.

When the next prototype is considered ready for use, it can be installed in the second Urban Neighbourhood. The experiences from the first Urban Neighbourhood can be used to design a plan for training and organizational implementation.

A third Urban Neighbourhood will be used to test if the system structure and range of options can be used in an Urban Neighbourhood which has not been able to influence the design and take part in the knowledge development. The results will, together with the results from the municipal survey, give the final input to changes

[19]Florence(85), Florence(87a)
[20]Florence(87b), Einarsrud and Granholm(86)
[21]Wegge(89)
[22]Floyd(84)

in the design, and form the knowledge basis for suggestions for local approaches to technological and organizational change in all the remaining Urban Neighbourhoods.

The approaches described requires a considerable amount of time and effort on the part of both developers and users. But, with the results of the present project in mind, we believe it is worth both the time and money spent.

References

Andersen, N. E. et al(1986): Professionel systemudvikling (Professional system development). Teknisk Forlag a/s, København 1986.

Avison, D.E. and G. Fitzgerald(1988): Information Systems Development. Methodologies, Techniques and Tools. Blackwell Scientific Publications 1988

Bjerknes, Gro, Pelle Ehn and Morten Kyng (eds)(1987): Computers and Democracy. A Scandinavian Challenge. Avebury, 1987.

Bjerknes, Gro(1989): Motsigelses-begrepet — et redskap for å forstå situasjoner i systemutvikling (The concept of contradictions — a tool for understanding situations in system development). Ph.D. thesis, University of Oslo, Department of Informatics, 1989.

Checkland, Peter(1981): Systems Thinking, Systems Practice. John Wiley & Sons Ltd., 1981.

Checkland, Peter(1985): Achieving "Desirable and Feasible" Change: An Application of Soft Systems Methodology. J. Opl. Res. Soc. Vol. 36, No. 9, pp. 821-831, 1985.

Einarsrud, Anne Kristin and Torill Granholm(1986): Spesifikasjon og implementasjon av deler av programmoduler for sykepleiere (Specification and implementation of parts of program modules for nurses). Master Thesis, University of Oslo, Department of Informatics, 1986.

Florence(1985): Gjensidig læring (Mutual learning). Report no 1 from the Florence Project, Department of Informatics, University of Oslo, 1985.

Florence(1987a): Læring som forutsetning for design (Learning as a precondition for design). Report no 2 from the Florence Project, Department of Informatics, University of Oslo, 1987.

Florence(1987b): Å implementere en idé — samarbeid og konstruksjon i Florence-prosjektet (To implement an idea — cooperation and construction in the Florence project). Report no 3 from the Florence Project, Department of Informatics, University of Oslo, 1987.

Floyd, Christiane(1984): A Systematic Look at Prototyping. in Budde et al: Approaches to Prototyping. Springer-Verlag 1984.

Gustavsen, Bjørn(1986): Förändring och infrastruktur. Arbeta, Människa, Miljø no.

4 1986.

Järvinen, Pertti(ed)(1987): The Report of the 10th IRIS Seminar. University of Tampere 1987.

Kaasbøll, Jens(ed)(1988): The Report of the 11th IRIS Seminar. University of Oslo 1988.

Pape, Tom Chr. and Kari Thoresen(1987): Development of Common Systems by Prototyping. in Bjerknes et al.(1987).

Reksnes, Astrid Marie N(forthcoming): Desentralisering — Brukerinnflytelse i Software Utvikling. Utvikling av EDB-systemer for Helse- og Sosialtjenesten i Oslo Kommune (Decentralization — user influence in software development. Development of computer systems for the health and social services in the municipality of Oslo). Master thesis, University of Oslo, Department of Informatics, 1990(hopefully).

Wegge, Daniela(1989): On Experimental Prototyping in User-Oriented System Development. in Bødker(ed)(1989): The Report of the 12th IRIS Seminar. University of Aarhus 1989

Weinberg, Gerald M.(1971): The Psychology of Computer Programming. Van Nostrand Reinhold, New York 1971.

Yordon, E. (1982): Managing the Systems Life Cycle. Yourdon, Inc. New York, 1982.

Øgrim, Leikny(1987): Decentralized System Development and Local Trade Unions. in Järvinen, Pertti: The Report of the 10th IRIS Seminar. University of Tampere, Finland, 1987.

Øgrim, Leikny(1988): Decentralized System Development: Women's Fortune? in Tijdens et al.(1988): Women, Work and Computerization: Forming New Alliances. North Holland, 1988.

Aardal, Kristin(forthcoming): Desentraliseringen i Oslo Kommune sett fra en bydel – en analyse av situasjonen i Søndre Nordstrand, forslag til mulige og ønskelige endringer. (The Decentralization of the Municipality of Oslo, Standing Point the Urban Neighbourhood of Søndre Nordstrand – Analysis of the Situation, Proposals for Feasible and Desirable Changes). Master thesis, University of Oslo, Department of Informatics, 1990(hopefully).