Design of Information Systems: Things versus People

Susanne Bødker
Computer Science Department
Aarhus University, Denmark

Joan Greenbaum
Computer Information Systems Department
LaGuardia College/City Univeristy of New York

Abstract

Information Technology is usually designed using traditional system development techniques and emphasizing conventional management objectives that focus on the information rather than the people in a workplace. This chapter uses research from a gender perspective that highlights the ways that office systems can be designed with people in mind. It then applies the gender perspective to explain why Cooperative or Participatory Design can be used to enable system developers and office workers to work together to design applications that better support working practices.

The terms Information Technology and Office Systems are heavily balanced in the direction of focusing on *things*, instead of looking at people at work. Both the methods used to produce these systems and the actual computer systems in use, reflect a strong bias towards looking at offices as if they were collections of systems that could somehow be solidified into a concrete object called information technology. In fact information technology's acronym, IT, speaks loudly to us about its focus. More than a century ago, Karl Marx described capitalism's fascination with quantifiable things. This fascination, which he called commodity fetishism, could be applied to the world of information technology, for the production of computer systems often looks like a process that develops, and sells information in packages of technology. But the fault with this fetishism of information technology lies not only within the realm of capitalism as Marx described it. This bias towards the relationship between things, such as information, rather than the relationship among people, also grows out of the traditions of Western scientific thought that sweep social issues to the side of replicable, quantifiable and provable facts. Embedded within scientific work, as Evelyn Fox Keller points out, are centuries of practices that push women's concerns with social interaction outside of the province of "correct" scientific thinking (Fox Keller, 1985). It is within this scientific tradition that the methods and practices of office system design were born.

In this chapter we will be discussing the ways office systems are developed and in doing so, we will look a little at the background of this development process and then examine some of the ways that it is changing and we believe, can change. First we discuss what is wrong with IT from the perspective of the systems approach. Here we find that the standard system development methods support traditional management objectives of quantitative efficiency often at the expense of the people in the workplace. Next we look at how research from a gender perspective can give us ideas for how and why applications can be designed to better support the way people work. And lastly we take up the issue of cooperative or participatory design, suggesting ways that system developers and office workers can work together to develop applications that show respect for the workers involved and result in office applications that better fit working practice. One can argue that cooperative design is useful from a number of perspectives including both an ethical and a pragmatic perspective. For example, ethically we would say that cooperative design can build workplace democracy and help support the human rights of office workers. Pragmatically, we can say that including office workers knowledge and experience in applications development results in technical applications that better fit working practices and are therefore more effective. Arguments about why or how a gender-based cooperative design strategy could be applied will, of course, vary depending on national and local traditions and problems. Here we present our analysis based on our experiences in this field.

As teachers, researchers and practitioners of system development we are caught up in the debate about how computer specialists and office workers (who system developers generally call the "users") can better interact with each other. This debate is not new, although there is definitely national and cultural differences in the way it takes place. The way we see things, the increased knowledge, and to some extent influence of office workers, has reshaped the debate. And furthermore managerial strategies have changed over recent years. Both from within the American perspective of Participatory Design (Namioka and Schuler, 1992), and from the Scandinavian approaches to system development (Greenbaum and Kyng, 1991), the issue is no longer only about whether these workers should be involved in computer system design, but also, *how* their knowledge and experience can be put to good use. While debate in the systems field has shifted to enfold the interests of the people who will use the systems, the methods and suggestions are often stuck in the historical frame of computer science and its reliance on formal problem-solving. In this sense, many methodological suggestions are still glued to a traditional management perspective, namely that of control over the system development

process. Furthermore they are, according to Christiane Floyd's (1987) framework, most often product oriented, rather than process oriented, emphasizing exactly the “thing” focus of Information Technology.

While our field is called system development, with an emphasis on office or computer *systems*, we prefer to talk about computer *applications*, in order to keep our eyes fixed on how the technology is applied, rather than on the technology itself. Our research is empirical as well as theoretical and in this chapter we will draw on some of our experiences in developing and researching workplace applications. We are interested in getting a closer look at how people work together within workplaces and on how developers work with people who will be using the applications. In doing this we have tried to apply a gender perspective to the way work is done. We would like to stress that what we are describing here is not the mainstream of computer development and use in Scandinavia or in the US. Rather, our examples are openings, telling us that there is realistic hope for qualitative changes in how applications are developed and used.

We begin with the idea that computer applications are socially constructed; that is designed and used by people, and not driven by some technological need. In applying a gender perspective we don't choose to focus on differences between the men and women in the field, but rather on the way that all system developers are taught to apply methods that, we believe, perpetuate traditional and stereotypical dichotomies between women's concerns with social interaction and men's so called "scientific concerns" with technology as a driving force. We believe that this is a false dichotomy that has driven a wedge between people and things, putting value on the latter and denigrating the former. Our use of a gender perspective tries to bring both values back into view with a hopeful eye toward developing computer applications that serve the needs of the people using them.

What's Wrong with "IT" ?

System development, and computer science, in general, is taught as a process of breaking problems down into manageable and controllable descriptions, a process which is the essential characteristic of what is called the systems approach. System developers learn to identify problems, solve them, and describe solutions. System development, in other words, is taught as a way of dividing and conquering the world, without getting

emotionally involved in it. Yourdon, the author of many books on system development methods has this to say about the process:

> "a design strategy that breaks large complex problems into smaller less complex problems and then decomposes each of these smaller problems into even smaller problems, until the original problem has been expressed as some combination of many small solvable problems." (1986, p.61).

In this way, office systems, like other computer applications, grow up as abstract models of reality. The notion of the world as a"system" is a way of reducing computer software and hardware as well as people to components, contributing to the overall system. These components are most often seen as describable in abstract terms. Another well-known system specialist describes the process this way:

> "The inversion of viewpoints occasioned by Structured Analysis is that we now present the workings of a system as seen by the data, not as seen by the data processors. The advantage of this approach is that the data sees the big picture, while the various people and machines and organizations that work on the data see only a portion of what happens. (DeMarco, 1978,p.49).

While many critics of office system design will argue that this focus on data flow is simply a mistake, the history of system development shows a long trail of similar "mistakes", too frequent and too important to be ignored. As early as 1965, Robert Boguslaw attempted to warn the system development community, who he called the "new utopians".

> "And so it is that the new utopians retain their aloofness from human and social problems presented by the fact or threat of machined systems and automation. They are concerned with neither souls nor stomachs. People problems are left to the after-the-fact efforts of social scientists." (1965, p.3)

The history of modern scientific thought takes its origin in the period when science drew away from the so-called 'irrationality' of women's knowledge and towards attempts to control nature presumably through reason and scientific procedures (Fox Keller, 1985). System thinking grew as the child of this scientific rationality (Ehn, 1989; Greenbaum & Kyng, 1991). It is within this scientifically oriented system development that the separation of people from things occurs, through incorporating formal sets of procedures for examining the things it identifies as data. Thus we believe that it as no accident or mere "mistake" that office systems frequently don't fit the people who are to work with them.

What is needed we believe, is a clear understanding of how these issues repeatedly come about, and what we, as office workers and system developers can do about it.

Applications development - a gender perspective

> "The most immediate issue for a feminist perspective on the natural sciences is the deeply rooted popular mythology that casts objectivity, reason and mind as male and subjectivity, feeling and nature as female. In this *division of emotional and intellectual labour*, women have been the guarantors and protectors of the personal, the emotional, the particular, whereas science--the province par excellence of the impersonal, the rational and the general--has been the preserve of men". (Fox Keller, 1985, p.2, italics added)

In *Reflections on Gender and Science (*1985), Evelyn Fox Keller points out how societally-shaped notions of men and women strongly influence the way that science is done. It also lets us see that as science split emotional from intellectual labour, the scientific arena focused more and more on things, putting people into the background. In her work she shows us, both historically and in practice today, how these notions or myths place women and men on binary poles where their differences are emphasized in the questions asked by scientists and the methods used to answer those questions. These binary poles, or dichotomies, shape the man's world as supposedly rational, objective, and quantifiable, while the woman's sphere is painted as emotional, subjective, intuitive and qualitative. As with all severe dichotomies, proving or disproving their validity takes us away from the real issues we need to tackle. What is at stake is the fact that they shape our consciousness and therefore our behavior.

Similar dichotomies can be seen in the field of system development. When we look at the procedures used to develop computer applications we generally see an overemphasis on seemingly objective criteria like hardware and software evaluation, with less attention paid to the so-called *soft* or subjective reactions of the people who use the applications (Greenbaum, 1990). As the quotes from Yourdon and DeMarco illustrate, computer system development is seen as a process of separating problems and cutting them up into manageable abstractions of reality. The development of office applications assumes that systems are so complex that they require *scientific study*; a form of examination that is somehow separate from daily life. This model building, which is supposedly rational and objective leaves little room for those more people-oriented activities that women are

socialized to be concerned with. In some ways the development of office systems can be seen as an embodiment of the separation of emotional from intellectual labor--a severing of the heart from the head. And super imposed on this frightening image is computer science's fascination, or fetishism with the head, and the abstract things it represents.

Our use of a gender perspective to the study of computer applications is not to bemoan the differences between men and women, for we feel that this unfortunately leads us back up the path where we find ourselves, once again, looking at "women on a pedestal" or "women as victims". Nor is it to complain about how the fascination with things and models has pushed social issues to the side. Instead we will choose to reframe the way we go about looking at offices in order to begin mending the head and heart and going about the process of designing systems that better suit the people who use them. In the following we look at some of the ways where current research gives us pointers towards bringing people back into the picture.

• Personal involvement in work

Here again we see Evelyn Fox Keller as a guide. In her book about the biologist, Barbara McClintock she talks about McClintock's research methods that let one get a 'feeling for', or 'listening to', the subject of the research (Fox Keller, 1983). Barbara McClintock marked her research with a strong belief that one should be 'letting the material speak to you'. Instead of descending on a research topic with a set of preconceived categories, McClintock chose to think of herself as being part of the material. For her, scientific work was living and being involved in the subject matter, or what she referred to as 'getting a feeling for the organism'. As system analysts we have tried to work in a similar way. We start from our personal involvement in our work and the way our work interrelates with others. For us applying a gender perspective means both getting involved in the workplace and looking closely at the people-to-people interactions taking place. As system analysts, we can not, of course, ignore the technical dimensions involved in developing office applications, but we can, however, put these technical issues within the context of the social setting.

• Complexity of office work

Elinor Wynn, a linguist took a similar approach in her study of clerical workers (Wynn, 1979,1991). Noticing that many computer systems were designed to be "idiot proof", as if clerical workers were somehow not very knowledgeable about their work, she undertook a study of clerical workers that highlighted the deeply complex and socially interactive nature

of their conversation. Her work showed how seemingly "simple" office conversation is rich in problem-solving and extremely important for getting work done. For system analysts Wynn's work illustrates how their reliance on scientific detachment may have pushed a focus on the information flow rather than on communication between people.

Wynn's work and that of many other researchers who study clerical work (U.S. Office of Technology Assessment, 1985), help us look behind the false assumptions concerning women in the workplace. As other chapters in this book illustrate, office work is often seen by outside 'experts' as a series of trivial tasks done by women. In the eyes of many management consultants, women's work is almost invisible. The most stated argument used to support the introduction of office systems is that of efficiency--making office output faster and cheaper. When system developers begin with the assumption that this work is trivial, these so-called efficient systems often end up causing more and more stress and serious physical damage to the women involved. Recent studies indicate that, among other issues, the emphasis on faster keyboards and more printed output has resulted in painful injuries like carpal tunnel syndrome and, in some cases, levels of stress that compare with those found among air traffic controllers (Labor Institute, 1988). Certainly the pragmatic argument can be made that information technology that results in injuries and stress is not useful for increasing overall efficiency, nor does it result in better service. By recognizing the complexity and richness of office communication and interaction, system developers and office workers can help stir the essential ingredients of communication into the way office systems are developed, highlighting both the ethical need for more humane systems and the pragmatic concern for applications that actually support work practice.

- **Changing situations and shared knowledge**

Lucy Suchman's *Plans and Situated Actions* (1987) gives us another starting point for refocusing the design of office applications. Her work shows that human actions are not always guided by clearly defined plans, but are based on actions within specific situations. While office applications are often designed with the former in mind, Suchman's research reminds us that applications need to take account of how people react and exchange information depending on the situations they find themselves in. Thus a system designed as if information always flows from one department to another, may break down when clerical workers in one department find that they have to go around the system to get things done their own way.

Another guide post in finding our way out of the gender stereotyped traps of traditional system development, is the recognition that we need to find ways to better respect *shared knowledge* in addition to the more generally accepted *authoritative knowledge*. Lucy Suchman and Brigitte Jordan (1989) define authoritative knowledge as "knowledge that is considered legitimate, consequential, official, worthy of discussion and useful for justifying actions...". System developers and managers present tools like data flow diagrams and decision charts as authorative knowledge. Often when the authoritative knowledge of system developers clashes with the shared knowledge of office workers, the office workers informal information gets buried within the "legitimated" authoritative power structure. In recognizing the importance of shared knowledge and by acknowledging the significance of situation-based activity system developers can better support people's working practices.

We believe that there is now more room to move application development in line with developing *tools for how people work*. This is obviously important from an ethical perspective, but now it is increasingly possible from a pragmatic perspective. In Europe and North America, for example, more and more office workers have some familiarity with computer applications. As they become familiar with applications they have also become more comfortable with voicing their concerns about systems that don't work. And increasingly managers are caught in a bind where they know that the data that comes from the Information Technology is not as good as the information the office workers have. There is room now for us to envision new ways of developing applications that go beyond the fascination with technical *things*. We feel that we are in a time where different paths are coming together giving both office workers and system developers the opportunity to set out in new directions.

Cooperative Design and the Integration of Emotional and Intellectual Labor

We don't believe that scientific rationality and authoritative knowledge have any better chance of withering away than Marx's predictions for the disappearance of the state under socialism. As computer scientists we know how well-grounded our discipline is in scientific and authoritative traditions. Yet we can clearly see that the ground is ready for system developers to begin to take seriously the issues that were uncovered in the preceding section. And in fact, as we will discuss, these are no longer theoretical starting

points, but rather issues that are being integrated into application design in both the United States and in Scandinavia. Three primary changes which have begun to occur include the need for system developers to:

- *focus on the workplace* and the actual practices of the people doing the work
- *involve office workers* at all levels in articulating their needs and expressing their concerns for what kinds of computer support they may need
- *develop new methods* that help system developers and office workers actively support ongoing social processes.

These issues are a few of the wide range of approaches coming together under the rubric of Cooperative or Participatory Design (see Greenbaum &Kyng, 1991). While Cooperative Design is not the mainstream of practice in the system field it does present ideas and practices that are being increasingly recognized. The cooperative approach means that computer application design activities begin and end with the people in the workplace. It includes training system developers to look at office workers as competent practitioners rather than invisible entities. Thus a payroll system that formerly had been designed to issue checks and documents automatically "without human error", now could be designed to allow payroll clerks to look up information and answer inquiries about problems like lost checks or incorrect tax codes. The resulting increase in quality of service could be viewed with equal, and perhaps greater importance, than the solitary focus on greater speed and increased output.

The importance of involving office workers in the design process has begun to take on added significance (see Friedman, 1989). Traditionally, users of computer systems were seen as the managers ordering the system. Increasingly, however, as the term "end-user" has crept into computer jargon it is apparent that both managers and system developers have begun to realize that the "end-user", or the person who actually does the work, needs to be included in the design of information applications (Grudin, 1991). Sometimes this means that secretaries and clerks are only asked to describe their work tasks, but from the perspective of Cooperative Design it increasingly means that these workers need to get involved and stay involved with the system developers as the new application is emerging.

Which brings us to the third point, that of finding new methods that help office workers articulate their needs and enable system developers to listen to them. It is in this area that we have had the most experience, and it is precisely this area that has been propelled along by help from a gender perspective. Here are a few examples of the types of changes we have seen taking place.

In a study of system development work at a large bank in Denmark (Bødker & Greenbaum, 1988), we conducted a series of workshops with system analysts to help them find out what improvements they wanted in their work. Among other requests was their frustration with only seeing a part of the development process, and being cut off from an understanding or overview of the project. They argued that if they had a clearer view of the whole project they would be in a better position to understand bank operations and therefore sort out some of the problems that the bank workers had with previous system design. As one woman pointed out: “It is embarrassing, when my friends ask me what I do, I tell them that I make systems for the bank. When they ask me how that affects them as customers in the bank, I must say I don’t know! I don’t know how the programs I make will work in the bank office. What I do know is how my modules interact with those programmed by others in the group”. In addition they urged that since development work was project or group oriented they needed to have a "feeling of belonging" to the group. This feeling of belonging, like their frustration with piece meal system development is not addressed in traditional systems literature, but is certainly a fundamental aspect of feminist research, for it recognizes the need for ongoing social processes that support the way people actually work.

In another project we saw that office workers who were given the chance to talk about their ideas concerning desk-top computers, gained a lot from the process and were able to express their needs for the type of hardware and software they wanted their organization to use (Greenbaum and Madsen, 1992). The project involved a dozen American office workers in a wide variety of titles from secretaries and editorial assistants to accounting clerks and editors. They participated in a series of workshops modelled after a technique called The Future Workshop (Jungk and Müllert, 1987), which encourages participants to speak freely among themselves and find constructive alternatives to their problems. The participants in the American office, like many others that have been involved in workshop environments found that their problems with the organization's current computer applications were not their own individual problems, but rather shared concerns that they could begin to do something about. Their use of this shared knowledge enabled them to

find ways to confront what had previously seemed to them like a wall of authoritative knowledge from the systems department (Greenbaum, 1991). They were able to listen to each other and get, in McClintock's words " a feeling for the organism". And from the stand-point of computer application development, they were able to better express their needs and concerns about the type of desk-top hardware and software they needed to get their jobs done.

In one case, in a Danish public office, we have seen a move from centralized system development to decentralized development where each branch office is provided with Fourth generation tools for their own use. These so-called Fourth generation tools are, among other things, computer languages that are designed for use by non-computer specialists. In this decentralization process a small number of workers from the branch office were trained as “computer instructors”. These instructors then act as both local system developers and continue to take part in the everyday business of the office (See Bødker et al. 1991). As with many systems this change was done to save money. At the same time, we see in the move an acknowledgement of the need for an anchoring of system development in the daily work of the office workers. Additionally, the office workers involved got a chance to learn new computer skills and to participate in designing the applications that they use. This case is still underway. It will be interesting to see if the office workers find it useful to act as computer instructors, and to what extent they feel overburdened by additional work. Much work remains to be done to find suitable ways of working with system development in decentralized work environments. (A research project focussing on these issues is outlined in Bødker et al. 1991).

To further support the increasing involvement of office workers in computer application development, system analysts have begun to use prototypes or trial systems to give workers a better chance to see and experience what the system will actually do. In one case, for example, with dental assistants, the system analysts developed computer prototypes that showed pictures of teeth so that the assistants could refer to the visual charts on the computer screen when they were working with patients (Bødker & Grønbæk, 1991a, b and c). As it turned out in this case, the system analysts had developed a "picture" of the mouth that was upside down from the way the dental assistants viewed patient's mouths. The problem was quickly solved since the prototype was merely a trial system and not the result of some long drawn-out traditional development project. Using prototypes lets workers get involved in very concrete aspects of application development, rather than relying on the abstract and often detached knowledge of system analysts. This

use of prototypes is gaining more acceptance in system development literature and practice. We would like to see more office workers get involved in testing prototypes, because we feel that this is an important issue for demanding more rights for office workers and for bringing about more people-oriented office applications. This may sound like a like a lot, particularly in rough economic times, but from our experience a common denominator among office workers and systems developers has been the desire to produce quality products and services.

While we could continue to go on offering examples of system analysts and office workers trying new ways of working together and sharing knowledge, we will end here and simply summarize our analysis of what is happening. Clearly the recognition of the need for office workers to get involved in the design and use of office applications is gaining popularity in the computer field. Over the last few years there have been a number of conferences, journal articles and trade publications addressing these issues (see for example Namioka and Schuler, 1992; CSCW'90 Proceedings; and recent issues of *Communications of ACM*). This is a good start. These changes are taking place because office workers are getting more knowledgeable about office applications and demanding to get involved. They are also occurring as system analysts and managers realize that the IT that they have developed often does not fit the needs of the workplace. At this point in time, moving toward Cooperative Design is more of a political process than a technical one. Changes in work organization and relative power relations of office workers, vary from country to country. We hope that this chapter begins to make a start toward ways that office workers and system analysts can envision more people-oriented computer applications and new ways of working together.

References

Boguslaw, R.(1965) The New Utopians–A Study of System Design and Social Change, Prentice-Hall, Englewood Cliffs .

Bødker, S. & J. Greenbaum (1988) A Non-Trivial Pursuit, cooperation in systems development, in J. Kaasbøll, (ed.)*Information Systems Proceedings* (IRIS), Røros, Norge,.

Bødker, S. and K. Grønbæk, (1991a). Cooperative Prototyping Studies - Users and Designers Envision a Dental Case Record System. In J. Bowers and S. Benford, editors, *Studies in Computer Supported Cooperative Work: Theory, Practice and Design,* Elsevier Science Publishers/North Holland, Amsterdam, pp. 315-332.

Bødker, S. and K. Grønbæk, (1991b). Cooperative Prototyping: Users and Designers in Mutual Activity. *International Journal of Man-Machine Studies*, 34, Special Issue on CSCW, pp. 453-478.

Bødker, S. and K. Grønbæk, (1991c). Design in Action: From Prototyping by Demonstration to Cooperative Prototyping. In J. Greenbaum and M. Kyng, editors. *Design at Work: Cooperative Design of Computer Systems*. Lawrence Erlbaum Associates, Hillsdale, NJ., pp. 197-218.

Bødker, S. et al. (1991) Computers in Context.-Report from the AT-project in Progress. *Proceedings of the NES/SAM conference*, Ebeltoft Denmark.

DeMarco,T. (1978) *Structured Analysis and system specification.* Prentic-Hall, Engelwood Cliffs, NJ.

Ehn, P. (1990). *Work-oriented design of computer artifacts*. Lawrence Erlbaum Associates, Hillsdale, NJ.

Floyd, C. (1987) Outline of a Paradigm Change in Software Engineering, in Bjerknes, G. et al., ed.: *Computers and Democracy – a Scandinavian Challenge*, Avebury, London.

Fox Keller, E. (1985). *Reflections on gender and science*. New Haven, CT: Yale University Press.

Fox Keller, Evelyn (1983) *A Feeling for the Organism: The Life and Work of Barbara McClintock*, San Francisco, W. H. Freeman.

Friedman, A, (1989), Computer Systems Developent, History, Organization and Implementation,Wiley, London.

Greenbaum and Madsen, (1992) Small Changes, in Namioka and Schuler (eds), *Participatory Design*, Erlbaum Associates, Hillsdale, NJ.

Greenbaum, J. (1990). The head and the heart, *Computers and Society* Vol 20, No. 2, June 1990, ACM, New York.

Greenbaum, J. (1991). The head and the heart revisited:Towards Participatory Design, in Lehto, Eriksson, & Katchum (eds), *Women Work and Computerization*, North Holland, Amersterdam.

Greenbaum, J. and M. Kyng, editors, (1991). *Design at Work: Cooperative Design of Computer Systems*. Lawrence Erlbaum Associates, Hillsdale, NJ.

Grudin, J. (1991) Interactive Systems- Bridging the gaps between developers and users, Computer, April, IEEE, Los Alamitos, CA.

Jungk, R. and N. Müllert, (1987). *Future Workshops: How to create desirable futures*. Institute for Social Inventions, London.

Labor Institute, (1988), *A Price for Every Progress*, (video), Institute for Labor Education and Research, NY.

Namioka, A. and Schuler, D. (ed.) (1992), *Participatory Design*. Lawrence Erlbaum Associates, Hillsdale, NJ.

Suchman, L. (1987). *Plans and situated actions: The problem of human–machine communication*. Cambridge: Cambridge University Press.

Suchman, Lucy & Jordan, Brigitte (1989) Computerization and Women's Knowledge, K. Tijdens et al., (ed).: *Women, Work and Computerization*, North Holland, Amsterdam.

U.S. Office of Technology Assessment, (1985) *Report on Office Automation*, U.S. Congress.

Wynn, (1991) Taking Practice Seriously, in Greenbaum and Kyng (eds) *Design at Work*, Erlbaum Associates, Hilsdale, NJ.

Wynn, E.(1979) *Office Conversation as an Information Medium*, University of California, Berkeley (dissertation).

Yourdon, Edward: *Managing the Structured Techniques*, Yourdon Press 1986