A Design of Ones Own: Towards Participatory Design in the US

Joan Greenbaum
Department of Computer Science/ Aarhus University, Denmark
and
City University of New York, LaGuardia College (on leave)

December 1991

Abstract

The Participatory design movement in the United States is, in many ways, growing out of the Scandinavian or worker-oriented design approach in Europe. This article discusses some of the roots of this movement and goes on to focus on ways that a 'home grown', or typically American brand of pragmatic system design can develop in the US.

The road toward participatory design in the United States has been under construction for some time now. Certainly by the middle part of the 1980's as most large companies grappled with the proliferation of PCs on employees desks, it was becoming clear that users of computer services needed more of a voice in the type of computer services they were receiving. This chapter will lay out some of the groundwork that has been leading to more user-centered system design, and explain how this can be used as a bridge toward building more participation into the system development process. Several issues will emerge here. The first is that user-centered design, while helping to pave the way towards participatory design, is not participatory in and of itself. The second addresses the issue of the differences between what has come to be known as the Scandinavian approach to design and the way participatory design in the US is emerging and is likely to develop. And the third is that while participatory design both requires and fosters workplace democracy, participation in the design process may not necessarily lead to workplace democracy. In fact, a central point of this chapter is to illustrate the ways that, I believe, American design needs to take on its own 'home grown' characteristics and to paraphrase Virginia Wolf, become 'a design of one's own'.

As the chapters in this book point out, participatory design is many things to many people. Yet there is a remarkable core to the ideas which has been built on common ground. Among the elements in the common core, are the ideas that: computer applications need to be better suited to the actual skills and working practices of the people using the systems; that work is a social activity involving the interaction of many groups of people; and that barriers between technical specialists and people using computer applications need to be broken down in order to build effective communication during the design process.

Participator design implies that workers as users of computer products should take part in the decisions that affect the system and the way it is designed and used. Since technology isnt developed in isolation, participation in decisions about technology also involves decisions about work content and job design. Seen in this broader context participatory computer system design needs to be part of an *integrated design* that looks at work organization, job content and the way technology is used to support these activities. This integrated design process also implies that system developers as technical specialists need to refocus their energies so that they can learn to rely on the expertise of the workers, and in effect 'de-expertize' or 're-expertise' themselves. (Dooreward and Regtering, forthcoming). System developers in an integrated and participatory environment would then take part in playing active roles in fostering and enabling people to use their knowledge to make decisions. But the step between fostering participation and enabling decision-making is grounded in a question of power.

1 Power Shifts: controlling the svstem development project

Participatory design represents a potential break with traditional approaches to system development. The break takes place when we recognize that full and active participation of users as workers implies a shift in power relations within companies. To address this issue more closely this chapter focuses on Information Systems (IS) within large organizations, particularly Management Information Systems (MIS). In the development of MIS applications the contradictions between the roles of system developer as management consultant and system developer as spokesperson or catalyst for user groups become quite clear. System analysts, whether they are employed by an organization ordering a system or work as outside consultants to a project, are usually hired by management to respond to a set of management defined objectives. Within the American context, the idea of a system analyst acting as an advocate for user rights is not yet an issue on the table of possibilities.[1]

This means that when system developers take an active role helping users become involved in defining and using computer systems they are moving into uncharted waters. The role of system analyst as management consultant or management technical specialist comes into conflict with that of system analyst as user-facilitator. For participatory design to work within the unstated assumptions of the American corporate world, these conflicting roles need to be spelled out and clarified. Movement from involving users in design to full participatory design means encroaching on some decisions that have traditionally been considered the prerogative of management. It also means looking at the concept of the integrated design of work organization, decision-making and computer support. In a sense participatory design opens up Pandora's box, for the questions we need to ask affect the whole organization and the way decisions are made within it.

[1]System analysts work in a variety of arrangements including work as in-house staff and outside consultants. Their work also varies depending on the extent of interaction they have with users (i.e. products for software houses where users are often not directly involved in the development process, or user-specific applications in application areas like banks). The point here is that regardless of the type of work done, most developers are employed by management to carry out management goals. This chapter focuses on MIS applications, for discussion of product development organizations see J. Grudin in this volume.

To better understand this situation a brief look at the history of system development is useful. Following World War II some of the early methods and procedures for system development came from the field of Operations Research (OR). OR based procedures emphasized quantitative reasoning on order to facilitate coding complex programs. In the 1950's as software for large mainframe computer systems was laboriously cranked out, practices to frame from the newly emerging field of Management Science also began the boundaries of system development field (see Greenbaum, 1979).

The merging of Operations Research and Management Science was not an accidental marriage. In the early days software development was extremely labor intensive and prone to error. System analysis and design emerged as a series of techniques that could place clear-cut standards on the way software was to be produced, including contractual arrangements to control the stages of development. Thus the system life cycle was defined as a series of linear steps from feasibility study through implementation and documentation which could be seen by MIS departments as isolated project parts to be defined, administered and controlled. Interpreting this from the labor process perspective of Braverman's book on *Labor and Monopoly Capital*, one could say that computer systems were in fact designed to further divide workers and in many cases take away the skills that they had (Greenbaum, 1979, Kraft, 1978). Viewed in this way the system development process has not only excluded users from the development of systems, but has done so by design (Noble 1977).

The steps in the traditional life cycle approach were originally developed to control large scale projects yet the emphasis on *control* has remained a cornerstone of the profession. So much so, in fact, that if one were to read through the leading textbooks on System Analysis ad Design today one would still find that the themes of dividing the project into stages and controlling each stage continue to dominate the material. While newer textbooks introduce chapters on prototyping and inject some sections on the 'human factor', most material still focuses on procedures for meeting management needs for control over project costs time-frame requirements. In short, how the system might suit the needs of people working within the organization has been secondary to management requirements for control over the project.

As early as the 1960's some observers began to argue that the life cycle approach and formal procedures were unresponsive to human needs. In 1965, Robert Boguslaw asserted that system developers were "concerned with neither souls nor stomachs", resulting in the fact that "People problems are left to the after-the-fact efforts of social scientists" (p.3). Today close to thirty years after Boguslaw's warning the tools and techniques taught to new system developers still follow the procedures for control outlined in the textbooks. In practice, the patterns of behavior among system developers and managers, for the most part, mirror the textbook methodology, or at least try to sound as if they do (see Friedman, 1989). System Analysis and Design as practiced within the confines of most Management Information Systems departments looks a lot like general management practices with some specific techniques thrown in to account for the difficulty of controlling software projects.

Yet over the last thirty years the pages of management and system journals have been peppered with articles bemoaning the fact that so many systems don't work or fail to do things that both managers and users expect them to do. In addition to critiques within the system field, social scientists who have studied people at work point out that many computer systems adversely affect the performance of the workers using them. Most work places seem to have stories that confirm these findings-stories about workers having to do extra tasks just to make the data fit the system or about people spending long hours trying to print out, letters on printers that are incompatible with their work stations.

General dissatisfaction about workplace systems seems to have reached a feverish peak in the mid 1980's when MIS departments began to complain more loudly about taking the blame for systems that did not work (Friedman, 1989). Some changes occurring during the last decade may help propel both MIS departments and users toward the need for some form of participatory design. The proliferation of PCs throughout many organizations means that many formerly passive users have begun to ask not what they could do for the MIS department, but what the MIS department could do for them. In fact, as many workers become more familiar with using computer hardware and software they no longer think of themselves as the silent majority. Thus the fact that MIS departments perceive that they are losing some control combined with the realization that users are getting more knowledgeable can lead us in the direction of practices that involve more participation from all who are affected by changes that occur when computer applications are installed or updated.

2 From User-centered to Participatory Design

In the mid 1980s research in the area of user-centered design pointed out the need for applications that were not just user-friendly, but rather were more deeply rooted in the practices of people using them. The user-centered approach attempted to bring people back into the picture, putting emphasis on the need to develop systems that worked in practice, not just in testing. During this period studies from the field of human-computer interaction have played an important role in bringing the social sciences and humanities into the formerly quantitatively-oriented system development process (Norman, Draper J986). The focus on user-centered design has been important for both raising the issues of social interaction in the workplace and for developing new perspectives to frame the way systems, particularly human-computer interfaces, could be designed. Yet discussions about user-centered design arising mainly within product development organizations have not tackled the thornier issues of control in the workplace nor have they directly addressed decision-making in a MIS environment. The extent to which an American participatory design movement can openly and clearly take these next steps, will in my view, determine the success of building a bridge from user-centered to participatory activities.

As discussed in other chapters in this book, Scandinavian approaches to system development offer ideas and examples that could be applicable, with some modification, in North America. In particular, the approach called Cooperative Design places emphasis on cooperation between system developers and users and focuses on the cooperative nature of work. *Design at Work* (Greenbaum & Kyng, 1991) lays out an approach to cooperative computer system design. It outlines and develops a rationale for using techniques that fully involve users in the design process. These techniques stress the need for system developers to learn from the experiences of people using computers, not just from formal system descriptions of work. In addition, cooperative approaches argue that workplace language and daily experience of users need to be placed center stage in an effort to enable users. For enabling users implies not just using their experience, but creating and fostering an environment where they can feel empowered to express their ideas.

The background for the work on Cooperative Design grew out of both the need for more user-centered design and out of the Scandinavian worker-centered approaches to system development (Bjerknes, Ehn, Kyng, 1987). While its origins lie in Scandinavia, Cooperative Design provides some necessary ingredients for U.S. system developers and users to begin the active and enabling process of working together. In this way Cooperative Design sets the stage for more participatori practices in both the U.S. and Scandinavia.

3 Building Participators Design in the U.S.

I would argue that the time is ripe for laying the groundwork for an American, 'home-grown' participatory design movement. This does not mean that through design we could alter the power relations discussed in the second section of this chapter, but rather that we could encourage more active involvement of users and developers in the design process. Here are some of the events and situations which, I believe, we can use to build strategies that foster a Participatory Design environment.

Management strategies—team work

Over the last several years management journals have focused on shifting managerial strategies, particularly those that emphasize the importance of team work. Economists have also addressed this issue in their discussions about the global challenges facing US industry. While the challenges and changes reflect a broad spectrum of opinion, the majority of the discussions center around the ideas that team work is essential in virtually all jobs and that enhancing communication is a co-requisite for coordinating team work. The new managerial focus on team work and communication opens the door for developers and MIS departments to introduce ideas like those of Cooperative Design that include issues of work organization and computer support. For as managers worry about the ways that team work can be encouraged, participatory design techniques suggest concrete activities for applying these principles in the design of computer applications.

This is not to say that Participatory Design answers managements' problems, nor should it. Rather, it is a way for system developers to get in the door with strategies that indeed increase the likelihood of more worker participation, and offer concrete suggestions for designing systems that might better fit the working environment. Since most system projects start as a series of management objectives, it may now be more possible to make arguments that management is likely to hear. In particular, given the shift in management theory, system developers can argue that participatory design teams can play a role in fostering communication within organizations. These type of arguments do not, of course, solve the contradiction that system developers are caught in as they try to meet management objectives and worker needs. But, as the experiences outlined in this book indicate, Participatory Design could help create an environment where system developers and users of computer applications can learn to develop systems that better suit the way work is actually carried out. And it can be used as a wedge to develop frameworks that look at the whole context of work organization and build towards designing future work situations that reflect the needs of the people working in them.

Multi-cultural work groups

Another aspect of the current American experience grows out of discussions about the composition of the workforce and the importance of recognizing multi-cultural diversity. This movement, starting in the schools and slowly spreading to business organizations, stresses that we recognize that the workforce is made up of people from many cultures. Social scientists who study multi-culturalism argue that this diversity can be used to enrich workplaces. On a very concrete level, multi-cultural pluralism lays out the arguments that support the need for participatory design. This movement recognizes that a workforce of people from many cultures not only speaks many languages, but may make differing assumptions about the nature of the work they are doing. The need to voice unspoken assumptions and make cultural pluralism a reality is a critical realization for American management. If taken seriously it could take the issues of civil rights to a new level, saying that civil rights in workplaces require acknowledging and respecting differences.

Obviously, the degree to which multi-cultural pluralism gains strength in the workplace will depend on political struggles that take place outside of work.

At the same time that pluralism is being talked about in educational circles, courts in the U. S. have been moving away from worker rights. Yet as long as pluralism is being discussed, possibilities exist to convince management that techniques like those of Cooperative Design offer a way to bridge language and experience gaps. Emphasizing group process, experience and workplace expertise, participatory processes could eat away at the old way of doing things. Specifically, projects that emphasize cooperation and group process are a way to shift the discussion away from management's narrow focus on control.

4 A Pragmatic Approach

Participatory design, as described in other chapters, involves process techniques. Some might argue that these process-oriented approaches may seem overly optimistic, yet they do provide a basis for system developers to make arguments that place their ideas on the list of possibilities that management is likely to consider. Experiences discussed in this book indicate ways that this is taking plate. As Friedman (1989) points out, while some managers remain wedded to the issue of controls others have recognized that traditional system techniques fall far shop of developing systems that work beyond their implementation phase. On a very basic level, many managers are ready to listen to new approaches because they are simply tired of taking the blame for bad systems.

Some advocates of participatory design have worried that the lack of a strong union movement would hamper its applicability in the US. Certainly the high degree of unionization in Scandinavia, coupled with legislation that allows for worker discussions about technology, has helped build a base for participatory ideas in those countries (Greenbaum & Kyng, 1991). I believe that while it would be desirable to have that level of worker support, it is, within the current political environment, not a likely reality for American workers. Indeed by the end of the 1980's American union membership had fallen below 17% as traditional union strongholds in large industry lost jobs. Since white collar jobs, particularly Management Information Systems departments has never seen significant unionization, it would be useless to assume that union activity is a prerequisite to a participatory movement. Rather, I would argue that strong American opinions about the value of democracy and a long history of emphasizing process in both government and education could form a base for building participatory projects.

As in most other movements in American history, pragmatic approaches carry some weight. Participation of computer users in development projects offers pragmatic possibilities for management, users and system developers. For managers it offers the possibility of getting out of the hole of being blamed for projects that do not work; for users it clearly offers the opportunity to expand on their knowledge of the workplace and indeed to feel better integrated within it; and for system developers it offers the chance to build systems that work better. Like the civil rights movement and the women's movement, American activities may start with what seem to be rather idealistic goals, but once the economic roots of these ideas spread out, the possibilities for building on them grow larger. Of course there are set backs and side turns, but the process of involving more women and minorities in American workplaces has grown over the last thirty years. Similarly participation in technological decision making will take some time to take hold. The framework of beliefs is in place, and if introduced as both democratic and pragmatic, the ideas could spread out and grow.

The common ground for pragmatic workplace politics is clearly in place. The availability of 'off-the-shelf' packages and techniques for rapid prototyping provide evidence that the time frame of the traditional project life cycle can be shortened tremendously. Additionally, users are increasingly knowledgeable about computer applications, and certainly more vocal about their likes and dislikes. Even managers intent on maintaining tight control over project development have begun to see that the age of building 'idiot proof' systems may be coming to an end. Also more sophisticated software applications require more sophisticated users, making it in management's interest to work more closely with people who are using the new systems.

System developers can respond to these situations with a variety of arguments that open the door to increasing user participation. Borrowing from the Scandinavian approach, we could say that participation helps users increase their skills and thereby increase the quality of the services they provide (Bjerknes, Ehn, Kyng, 1987). To managers bent on finding 'solutions' that increase productivity, there is an obvious link between enhanced service and greater productivity. Indeed economists have found it almost impossible to measure increased productivity in the white collar sector, so rather than beating the same drum, managers could be encouraged to see that enhanced user participation and even the possibility of users providing better service might be more efficient than simply creating systems that increase paper or screen output.

While to my knowledge, no cost studies of actual participatory design projects have yet been done, word-of-mouth reports indicate that involving users intensively and early in project development does not seem to increase project costs. And many argue that the increased time needed for these early group experiences, pays off in the likelihood that the project won't get bogged down toward the end when it is 'discovered' that the system doesn't meet some crucial workplace need. Many MIS system developers have said that they had little trouble convincing managers that some regular 'release time' was needed for users to get time off from regular duties in order to take part in design activities. As more and more of what gets done in companies begins to be recognized as part of participatory design, I believe that system developers will begin to swap success stories at an amazing rate. As noted in many conferences over the last few years, the stories of designing 'not by the book' have been a growing topic in hotel corridors and over coffee breaks.

I believe that the base exists to move from lip service to user participation to more active participation of users in the design process. More active participation, using the experiences discussed in this book, includes seeing users play an advisory role in decisions about computer support. And hopefully it includes watching users get a larger role in, at least advising, management about work organization and workplace environment issues. If we build on some of the experiences in this book we can clearly see that the *participatory* part of participatory design includes ways that: users gain more experience and knowledge about technology when they actively participate in project groups; system developers and users get better at designing and working with appropriate prototypes; users gain from getting 'hands-on' experience of trying out possible software; and system developers do leak from the expertise of users-learning that in fact results in more workable prototypes and products.

5 From Participation to Workplace Democracy...

As discussed in the introduction to this chapter, the step from participatory design to workplace democracy is not automatic, for it involves serious shifts in control over derision-making. The movement from active participation in an advisory capacity to actual decision-making is a very big step. And the shift from system developers as MIS experts to more neutral facilitators is not so easily accomplished. Many system developers have found that while management might encourage them t.o actively enlist user participation that mandate does not include user decisions over the final system.

Americans are fed a steady diet of products and ideas that include the word 'democracy', yet within the confines of the workplace the concept is rarely if ever addressed. For at least one-third of every day, most Americans enter their work places and abandon discussion of this basic constitutional right. As in the discussion about management control over system projects, this lack of workplace democracy is no accident. For application of democratic ideals within the working environment would mean that on some level evervone could participate in decision-making—decision-making that in the American framework is clearly management's prerogative.

This topic is broader than the subject matter in this book. While my hope is that users as advisory participants can lead toward users as active participants in decision-making, to date we have little to judge how this might happen in American workplaces. Certainly legislation of the sort that exists in Scandinavia would be necessary to help protect the ergonomic environment and to give workers the right to participate in decisions that affect their future workplaces—decisions that need to include the type of technology used and the applications for which it is used. The road from participatory design to workplace democracy can be built, in part, on the base of successful participatory design projects. This book and the experiences that people tell about the success of participatory approaches work toward building that road.

Acknowledgements

This article grew out of a talk that I gave at the closing Plenary session of the Participatory Design Conference PDC'90, and will appear as a chapter in the forthcoming book entitled *Participatory Design*, Namioka, Aki, and Schuler, Douglas (eds), (Erlbaum Associates, Hillsdale N.J). It is built mostly on my reflections about working with Cooperative Design in Denmark, and in part on my experiences as a teacher and practitioner in the U.S. I want to thank the editors for lending me a copy of the tape of my talk and for helping me shape it into this chapter. Jonathan Grudin was wonderfully helpful in clarifying arguments and marking muddled sentences. My Danish colleagues have also been helpful, as always, in pointing out the *fuzzy* parts of earlier drafts. Any remaining fuzzy ideas are, of course, my own.

References

Bjerknes, Gro, Ehn, Pelle & Kyng, Morten (eds) (1987). *Computers and Democracy—a Scandinavian Challenge.* Aldershot, UK: Avebury.

Boguslaw, Robes, (1965). *The new utopians—A Study of System Design and Social Change.* Englewood Cliffs, N.J.:Prentice-Hall.

Braverman, Harry, (1974). *Labor and Monopoly Capital: The Degradation of Work in the twentieth century*, NY: Monthly Review Press.

Dooreward, Hans & Regtering, Harrie, (forthcoming). *Integrated Design.*

Friedman, Andrew, (1989), *Computer Systems Design: History, Organization and Implementation.* Chicester, UK: Wiley.

Greenbaum, Joan, (1979). *In the Name of Efficiency, Management Theory and Shopfloor Practice in Data-Processing work.* Phila, Pa.:Temple University Press.

Greenbaum, Joan & Kyng, Morten (eds) (1991). *Design at Work: Cooperative Design of Computer systems.* Hillsdale, NJ: Lawrence Erlbaum Publishers.

Kraft, Phillip, (1977), *Programmers and Managers, The Routinization of Computer Programming in the US.* NY: Springer-Verlag.

Noble, David, (1977). *America by Design, Science, Technology and the*

Rise of Corporate Capitalism, NY: Knopf.

Nolan, Donald & Draper, Stephen, (1986). *User Centered System Design: New Perspectives on Human-Computer Interaction*, Hillsdale, NJ: Lawrence Erlbaum.