Published in Journal of End User Computing, 1999, Issue 4, pp. 3-13

THE REALITY OF USER-CENTERED DESIGN

Susan Gasson
Binghamton University, State University of New York, Binghamton, NY, USA
Email: sgasson@binghamton.edu

ABSTRACT

It has been suggested that user-centered design approaches may be the key to matching the development of organizational information systems to a firm's business and work requirements. The information systems literature on this topic centers upon methods to support user-centered system development: such approaches assume that the participation of relevant stakeholders will ensure an appropriate design outcome. The reality of this assumption is examined through an interpretive case study. The project team and system development approach were structured around the use of user-centered design methods, yet the initiative failed because of the poor integration of users' interests. It is concluded that the failure mechanisms observed may be general to most information systems projects: issues more fundamental than the employment of a particular methodology need to be addressed, to achieve user-centered organizational information systems.

INTRODUCTION

Empirical research studies have suggested that user involvement in information system development is related positively to user perceptions of system usefulness (Amoako-Gyampah and White, 1993. Baroudi et al., 1986). This does not mean, however, that user involvement in information systems development is a necessary condition of success. Cavaye (1995) argues that there are also projects where users did not participate in the development but which are nonetheless successful and that the existing body of user participation literature is fragmented, presenting inconclusive results. It has been argued that it is not so much user participation, as user *involvement*, defined as that "state of psychological identification with some object, such that the object is both important and personally relevant" (Kappelman & McLean, 1992), which leads to success (Barki & Hartwick, 1994). Although users may be encouraged to participate in development processes, this does not mean that users are truly involved as equal participants in those processes: implicit power-imbalances and assumptions are embedded in traditional methodologies for IS development which prevent users being involved as co-agents in design (Beath and Orlikowski, 1994. Markus & Bjorn-Andersen, 1987). An ambivalence exists between the recommendation for "strong user involvement" in a particular development methodology and the degree to which users can be expected to be true co-agents with IS developers through the procedures and design mechanisms of that methodology (Beath & Orlikowski, 1994). In response to perceived inequalities of involvement, many 'alternative' methodologies have been proposed, based upon a user-centered approach; such approaches are designed to ensure more significant user involvement in the design process (e.g. Avison & Wood-Harper, 1990. Checkland, 1981. Mumford, 1983). The weak point of such studies is that they assume that the use of a user-centered IS development methodology will ensure significant user involvement: the quality and extent of user-involvement outside of the carefully-managed social contexts in which such approaches are tested is not examined.

The IS development literature does not really tell us much about how users are involved in system development practice. Studies of methodology do not often consider user-involvement as separate from the use of a particular development methodology; where they do so, they refer to user-involvement with a single question which asks if users were involved (e.g. Sumner & Sitek, 1986).

This paper presents a case study of a user-centered IS development project, to examine whether there are properties of IS development environments which undermine significant user involvement.

INTRODUCTION TO THE CASE STUDY

The case study investigated the design processes involved in a research project to investigate the design and use of a computer-based learning environment system for students at a UK University. The design was highly innovative: no previous system of this kind had been implemented when this project was initiated, in 1991. Students would be able to access sources of teaching material from a central data repository, be able to interact with each other and with members of staff over long distances by placing a message in a "mailbox". Students would be able to request help or submit comments on course-related topics, be able to place their own subject-related information in the central data repository for access by other students or staff, and would have access to a support-network of other students and staff, even though physically remote from the University. This research project was therefore focussed upon achieving an appropriate information system design: project deliverables were not clearly specified at the start, as a major part of the project was the exploration of what form this type of system could take, which is why a user-centered design approach was selected by the research Project Director.

The main interest of this case study was to understand the nature of the user-centered design process in practice and to investigate the constraints of this approach. A process model is distinguished from a factor model in that the former portrays ISD as a dynamic social process where the latter demonstrates a relationship between predictors and outcomes, without explaining how or why the predictors and outcomes are related (Newman & Robey, 1992). Thus, process models are most appropriate to the examination of ISD issues from an interpretivist perspective (c.f. Walsham, 1993), where knowledge is seen as a social construction by human actors. Factor models are more appropriate to a positivist research perspective, which attempts to measure the extent of an assumed dependency relationship between predictors and outcomes (Newman & Robey, 1992).

Because of subjects' sensibilities, I have refrained from identifying the University at which this study took place, and from using individuals' names in the descriptions that follow. This is not to imply that I view the problems which this project suffered as pertaining to the individuals involved; on the contrary, the analysis of this design project raises some important issues for structural and integrative constraints upon user-centered design.

The Context Of The User-Centered System Design Project

The constitution of the project team over time is shown in Figure 1. It was originally intended that the IS research and development project should have a duration of three years, from January 1992 to December 1994, during which time the form that the new information system would take would be jointly explored by a cross-disciplinary team of organizational psychologists, who would investigate, evaluate and represent user requirements of the system concept, and information technology software developers, who would demonstrate how such requirements could be fulfilled using new forms of information technology. The project terminated after a period of two years, when the Project Director was unable to raise further funding.

The design team at most points in the project consisted of four individuals, two technical developers and two organizational psychologists, reporting to the research Project Director who was himself an organizational psychologist. The membership of the core team varied over time, but its constitution did not: as one member left the team, another was recruited from the same discipline. The research unit technical systems manager was also involved in the project over its lifetime: initially in defining the project scope and later in providing technical systems support to the project team.

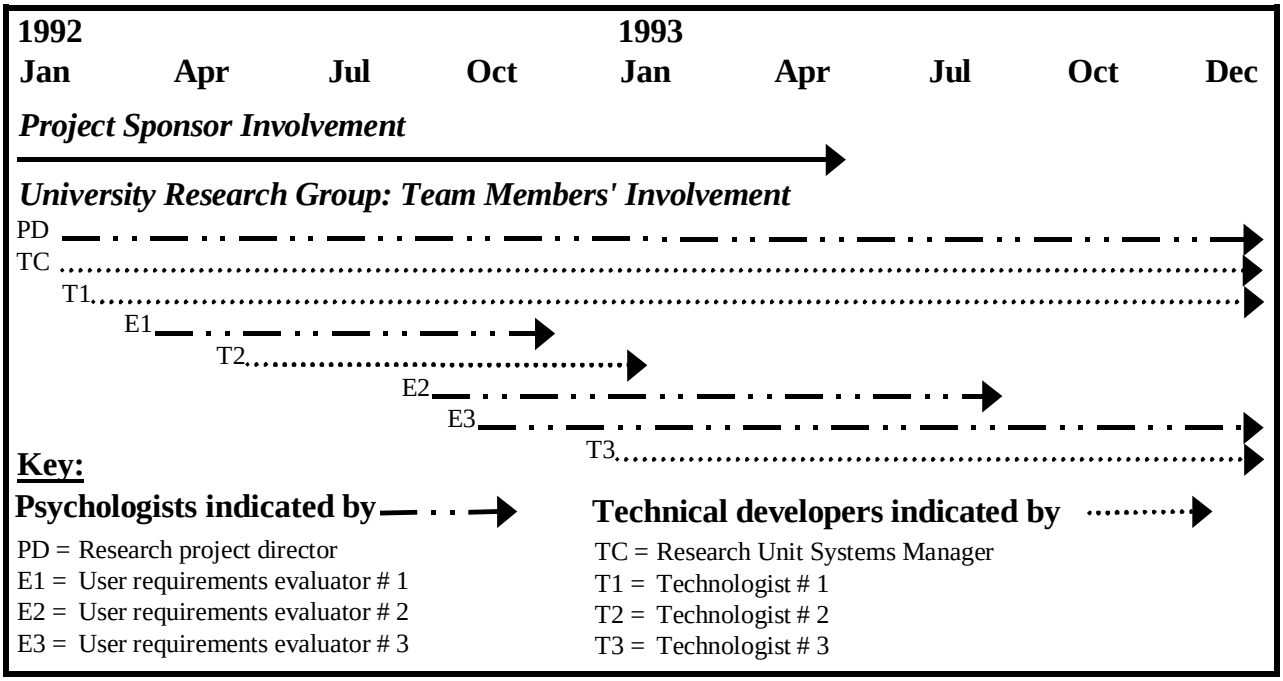

Figure 1: Involvement of Main Actors in Project, Over Time

Research Method

A set of semi-structured interviews were carried out with core members of the project in December 1993 as the project was terminating. Interview questions were developed, based upon the extent of adherence to the published process model (which was obtained beforehand) and upon the literature on user-centered design. It was not possible to interview all of those involved, as some of the external stakeholders were untraceable and one of the departing technical developers was alienated from the project to such an extent that he refused to be interviewed. Case study interviewees were therefore selected to represent continuity of perspective. Members of the core development team throughout its lifetime were interviewed: the Project Director, two organizational psychologists and two technologists. Interviewees were asked to describe the sequence of events during the design process. It was realized that there would be problems of post-rationalization (Giddens, 1993), so the critical incident technique (Flanagan, 1957) and data triangulation (Yin, 1994) between multiple sources of data were employed to understand the series of activities which constituted the design process, how these activities were approached, how they were interpreted by the design team, and what result was achieved.

Project documents produced by various members of the design team (research papers, project plans and information system requirements documents) were analyzed to interpret the process-models of design and the 'stories' and metaphors which the documents contained. Then the interview transcripts were coded, initially to identify commonality in critical incidents and descriptions of the process. A representative model of the actual (as distinct from intended) design process was derived from this coding. It was observed that the actual process model was very different from that shown as intended in project planning documents. Reasons for the difference between the two process-models (intended and actual) and an understanding of the constraints which had operated upon the user-centered design process were pursued by analyzing interview-transcripts and project documents, using a rigorous grounded theory approach (Glaser & Strauss, 1967). Taking the advice of Lowe (1996), the gerund form (ending in -ing) was used to label each identified theme, to "sensitize the researcher to the processes and patterns which may be revealed at each stage" (Lowe, 1996, page 8). The initial, open

coding scheme was gradually refined: codes were merged and some new concepts were added, with, as far as possible, triangulation between interviews used for core concepts. The analysis was hermeneutic in nature: that is to say that analyst tried to interpret themes and constructs as understood by the subjects of the study, rather than applying their own interpretation of them. This is not an objective process and issues identified will necessarily rely on the researcher's subjectivity, even if only in discriminating the significant from the insignificant. It proved possible to validate the process model and to follow up some of the themes identified in the grounded theory analysis through interviews with two of the original interviewees (one technical developer and one organizational psychologist), but it was not possible to obtain follow up interviews with all original subjects.

CASE STUDY FINDINGS

Commencement Of Project

The project commenced formally in January 1992. The project was instituted by the research Project Director, with help from the research unit computer-systems manager. Recruitment of staff started with the recruitment of a senior technical system developer, the most senior member of the design team in terms of research grade salary-point, which influenced both his own and other team members' perceptions of their relative roles. It was felt by other core team members that technical developer(1) had the strongest sense of project ownership. Shortly afterwards, the first user-requirements evaluator was appointed to the team. The first few months of the project were occupied with what all the team members interviewed called 'planning' activities: the determination of appropriate activities for user-centered design.

The Intended Process Model Of User-Centered Design

The system development approach was defined by a process model published early in the project; this is shown in Figure 2: it was defined by user-requirements evaluator(1), who had experience of working on a previous user-centered design project, jointly with the Project Director.

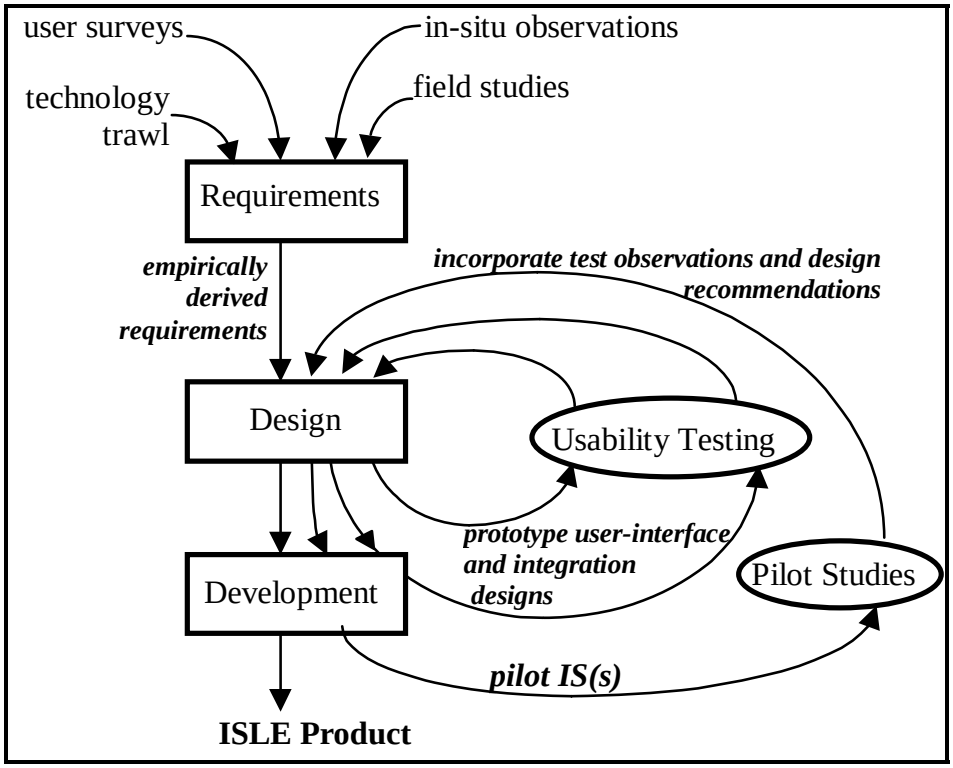

Figure 2: The Intended Process Model For The IS Design Approach

The system design initiative was intended to be both a research project, to investigate the learning environment concept, and a system development project: to develop a technical system which would support the innovative learning environment. The model shows a clear intention to drive the process around the needs of potential system users, where user requirements and evaluation of system use define the nature of the system 'product'. The outline process model was agreed by the project team and the project sponsor.

The Actual Process Model And Critical Incidents During The Design Process

In contrast to the intended process model, figure 3 shows the actual process of design. Five critical incidents were identified, which are discussed below.

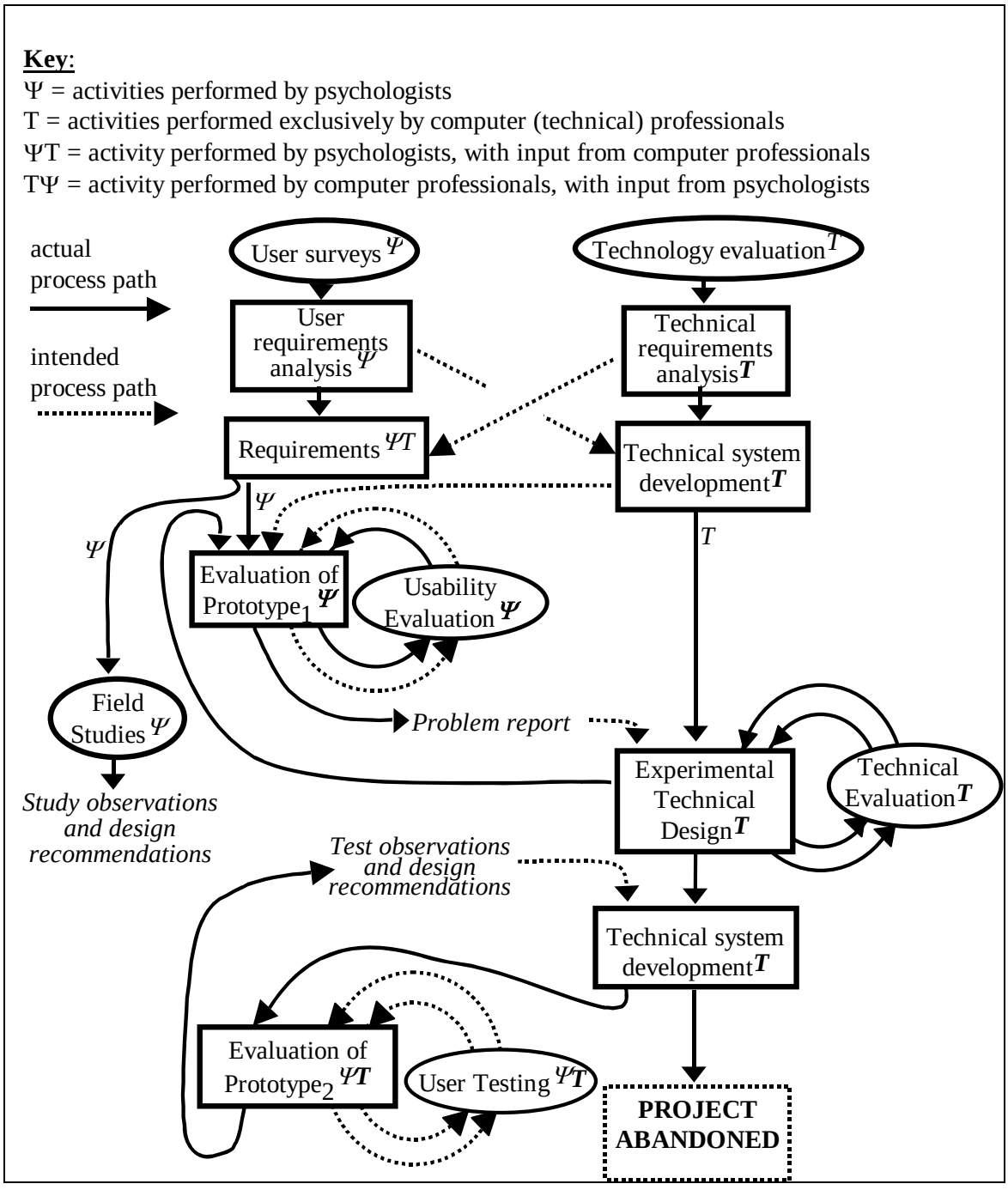

Figure 3: The Actual Process Model For The IS Design Approach

Critical Incident 1: Agreement Of Project Plan

The initial project plan, dated February 1992, was produced by the Project Director and the research unit computer-systems manager. This plan gives four main objectives for the first phase of the project, to last until end June 1992:

1. To plan the three-year project in detail.
2. To design an evaluation methodology for the introduction of technology-based teaching, with particular emphasis on the impact and usability of the technology.
3. To conduct usability studies on the sponsoring company's existing tools in this application area and to integrate the sponsor's technology with existing university distributed IT systems.
4. To initiate a development plan for the innovative learning store concept.

The involvement of the project sponsor, a multinational IT business systems supplier, required more formal planning and specification of intermediate deliverables than is usual for University research projects; as a consequence the next project plan, dated April 1992 was produced by user-requirements evaluator(1), who had experience of formal project planning. There were now four people involved in the core project team: the research Project Director, the research unit systems manager, user-requirements evaluator(1) and technical developer(1). The innovative nature of the project had also attracted a number of peripheral stakeholders from other research or teaching groups in the University, who were involved in design discussions as they wanted their interests represented in the design. This led to multiple, often unrealistic definitions of system goals.

A lack of commitment to formal project planning meant that detailed deliverables were never clearly agreed among the various stakeholders and that detailed project objectives remained unclarified. This was not perceived as undesirable by the research Project Director, as objectives in a research project are often emergent. But the lack of clear goals was exacerbated by changes in the project sponsor's ideas on how the new system might use the innovative information technology. There were political pressures arising from a restructuring of the sponsor's company, which meant that this project was perceived as under pressure to deliver a prototype technical system quickly. The project thus suffered from conflicting pressures for delivery: on the one hand commercial pressures from the project sponsors were demanding formalization of concrete deliverables from the project, leading to a concentration upon concrete technical system requirements, while on the other hand the experimental, emergent nature of the innovative system application required that system design alternatives be the subject of ongoing debate. The initial project plan attempted to satisfy both interests, while rejecting many of the more unrealistic expectations of external stakeholders.

Critical Incident 2: Marginalization Of User-Requirements Evaluator(1)

Differing perceptions of the need for collaboration in system design (and, indeed, in the definition of "system") led to a power struggle between user-requirements evaluator(1) and technical developer(1). User-requirements evaluator(1) assumed responsibility for project planning and at first attempted to collaborate with technical developer(1) in the design activity underlying information system definition as she conceived the core 'problem' of design to be the need for user requirements to drive the technical system design. Technical developer(1), meanwhile, had to engage in an intensive learning process (which was described within the project as a "technology trawl") and had also to appear to be delivering usable technology, in order to maintain credibility (in terms of technical expertise) with the project sponsor and other team members. He was unused to operating under conditions of such high uncertainty and responded by rejecting the reporting and collaboration formalisms of the user-centered design approach, as he saw such activities as a diversion from the technical focus of his work. User-requirements evaluator(1) responded by invoking formal task deadlines, over which she had control, demanding that technical developer(1) produce a technical

report: this "never got done". Technical developer(1) now erected a visibility barrier around the technical system by refusing to support user-requirements evaluator(1) in her use of the technology. For example, he failed to provide information on how the software, which he was evaluating technically, needed to be configured in order for it to work. If user-requirements evaluator(1) had been permitted to participate in the evaluation of alternative technologies, technical developer(1)'s learning process might have been exposed and his uncertainty regarding functional system requirements increased by considering complex user-requirements. User-requirements evaluator(1) therefore produced a user-requirements specification from field studies of *other* computer-based learning systems and interviews with students about their requirements of a learning-support system. Technical developer(1) produced a rival, technical system specification document which ignored the user-requirements identified from the field studies. As a result of increasing marginalization, user-requirements evaluator(1) resigned from the project.

Critical Incident 3: Delivery And Evaluation Of First System Prototype

Three new team members were now recruited: two user-requirements evaluators and an additional technical developer. Over time and with the arrival of new team-members, technical developer(1) was able to redefine the evaluators' work-roles. It is interesting to contrast user-requirements evaluator(1)'s description of her role with that of user-requirements evaluator(3).

> *User-requirements evaluator(1):* "The whole work philosophy is co-operative design, co-operative evaluation. The whole idea is highly iterative, going through cycles of trying it out, putting it to other people, going through redesigning it."
>
> *User-requirements evaluator(3):* "Evaluation is an ongoing process of looking at a product against both your idea of its requirements and the feedback that you have from the people using the product and then iteratively changing the product. … That's what I was taken on to do"

User-requirements investigation was by now seen as peripheral to the core problem of "design", by the technical developers and people without relevant technical knowledge were excluded from effective decision-making. The pejorative term "flower arranger" was coined to describe user-requirements evaluators by technical developers (it was even used to address an evaluator in my presence). Evaluators were not consulted about a major design decision, which would affect the timescale and complexity of the system design, while the opinion of a Ph.D. student and a technical Professor from another department, who just happened to be in the technical developers' office at the time of the discussion, contributed to this decision as they were "qualified to comment". The narrow system objectives defined by the technical developers constrained the variety of alternative design perspectives generated. Those alternatives which were generated were not exposed for view to the evaluators; they were validated informally and in private, so the integration of social and technical design perspectives, which forms the basis of user-centered design, did not take place.

This was legitimized by the appropriation of the term "design" by the technical developers. All team members now referred to the technical design as the "core" design: it was no longer seen as illegitimate by other stakeholders, including the Project Director and at least one of the evaluators, to exclude the evaluators from design processes. The technical developers had managed to define the role and work-activities of the evaluators in such a way that the latter no longer had any direct participation in defining system requirements. Even the language used had been appropriated by this point in the project: the user-requirements evaluators were now described as "evaluators", while the technical system developers were described as "designers". The evaluators could by now be directed, by the technical developers, to tasks which removed them from direct design participation, an avoidance in which they colluded. User-requirements evaluator(3) commented that:

> "The designers had rather more drive in that they could define the way the project seemed to be going rather more than the we could … we were trailing along behind them. It wouldn't be unfair

> to say that for most of the time the technical people just wanted us out of the way, so that they could get on with developing the product. …We sort of colluded with that as well, because there was this slight tension between the two sides, it was sometimes easier to just be doing something else. So both of us were doing other case studies which did not directly involve the designers."

The second user-requirements evaluator recruited to the team had previous experience of user-centered design and was not prepared to be marginalized from the design process. She responded with hostility to the technical developers' attempts to exclude her from design discussions; they responded with "polite antagonism". There was an increasing formality of communication between the two user-requirements evaluators and the two technical developers: team members were corresponding by email, when their offices were only 25 yards apart. Because of the changing nature of work-roles, this was now seen as a personality clash by the Project Director, rather than as a structural problem. User-requirements evaluator(3), who accepted his technically-proscribed role more readily (despite having a highly technical background himself), initiated informal discussions about the design, but reported that he might have no contact with the technical developers for several weeks. The Project Director attempted, at this point, to exert some control over the nature of the design process by insisting that the two technical developers produce a prototype technical system in time for evaluation in the context of a spring term Masters course.

The technical developers interpreted the nature and significance of a prototype very differently to the user-requirements evaluators: the prototype strategy was described by technical developer (3) with the words "you build two prototypes and throw one away". It was made clear by the technical developers that the first prototype was never conceived as an evolutionary basis for the design: they even gave this system the name "prototype minus-one". There appears to have been an emergent strategy on the part of the two technical developers that the first prototype was produced as a diversionary tactic to occupy the user-requirements evaluators while the IS professionals proceeded with what was referred to as the "real" design. It was easier to produce a working technical system from a technology which they had evaluated and discarded than from that which they were intending to implement. The user-requirements evaluators realized that they were being diverted in an attempt to keep them busy, but could do little other than go along with this.

Critical Incident 4: Withdrawal Of Project Sponsor

The project sponsor, a major computer equipment manufacturing company, decided to cease its involvement in the computer-support of academic and training systems and ceased funding the project in April of its second year. The project thus lasted for two years instead of the intended three years; this meant that project objectives had to be re-planned early in 1993 with the intention of redefining deliverables. This occurred soon after the delivery of the first prototype.

The role of the project sponsor had been disruptive to the initial design process, because of the commercial pressures which they exerted for delivery of design "products" and because of the changing nature of their expectations. Initially, the project sponsor had intended that some of their own experimental software would form the basis for the technical system design and that the project team’s involvement would lie mainly in designing an appropriate human-interface for that software and in the configuration and evaluation of the learning system concept in use, with real students studying a real course. However, the experimental software did not meet the sponsoring company’s expectations and the pressure on the project to produce software for a complete technical system became significant. Pressures to produce were highest on the two technical developers, who managed these pressures by concentrating upon technical experimentation to the exclusion of exploring shared, socio-technical objectives, refusing to collaborate in any way in a participative design process or to communicate what they were doing in detail. This response appears quite rational in the face of continually changing objectives and stakeholders: the pressure to achieve something was high.

Critical Incident 5: Delivery And Evaluation Of Second System Prototype.

A second technical system prototype was eventually produced, for the objective of user-evaluation. But the technical developers were affected radically by what they viewed as constant changes in project objectives - they responded by cutting themselves off to the extent that they ignored collaborative objectives completely and maintained no dialogue with the user-requirements evaluators. They were thus unable to perceive system requirements "changes" as a natural consequence of use requirements emerging from increased experience of the target technology on the part of the user-requirements evaluators, but interpreted them as irrelevant distractions, produced by stakeholders with no understanding of the "real" (i.e. technical) design issues. User-requirements evaluators did not see technical requirements definition as appropriate, as they were defining the context of use, while the technical developers continually referred to user-requirements as "user-interface requirements" and took the attitude that they were best placed to determine how the system would be used based upon the technical possibilities. They seemed unaware of any value in examining the *context* of use to determine structural system requirements. For example, the technical developers were uninterested in exploring what type of information students wished to store and access, or how, although these issues fundamentally affected the choice between competing technologies. The technical developers were unable to produce a usable system prototype, as they had little understanding of how such a system might be used in practice and the user-requirements evaluators were unable to specify user-*interface* requirements, as they had little understanding of the technology being implemented. The prejudices of the technical developers, who expected system requirements to be couched as technical requirements were thus confirmed.

An interesting side-effect of the technical developers' isolation was the way in which they were able to manage other team members' expectations of what they could produce, while setting their own agenda for the design. While the Project Director and the user-requirements evaluators reported that the technical people had not delivered prototypes when expected because they were overloaded with work, technical developer(3) stated that they had lost interest in the project because of continuing changes in direction and so they spent a great deal of time experimenting with interesting technologies rather than completing the design. The technical unit systems manager commented that, in retrospect, "user-interface issues" were undervalued, as the designers were more interested in technical experimentation. So the technical developers saw their role as defining both the human-computer interaction elements and the technical functions of the system and were demotivated to such an extent that they felt no compunction in delaying a usable system design to pursue other interests. They excluded the user-requirements evaluators from any role in the design, other than as neutral technology testers. This meant that the second prototype was not produced to be usable in any way which was of value to the project: the system interface was so poorly designed that it could not be used without personal supervision from a technical developer. The second prototype could not be used by students in any meaningful way. In the end, the team failed to achieve a coherent design which fulfilled the initial, user-centered design objectives and, when research funding was withdrawn because of restructuring at the sponsor company, the system design was insufficiently clarified and the technology insufficiently usable for the Project Director to be able to obtain further funding.

Strangely, towards the end of the project, a feeling of shared adversity appears to have brought the two sides closer together. The situation was also helped by the departure of the second user-requirements evaluator, as the remaining user-requirements evaluator was perceived as being much more technical (and therefore valued by the technical designers).

But the gulf between the technical and the use interests was still apparent, as shown by a comparison of the five ‘recipes for success’ in user-centered design obtained from interviewees:

Project Director: "defining the design problem clearly"

User-Requirements Evaluator(1): "investigating user requirements properly before attempting technical system design"

User-Requirements Evaluator(3): "investigating the problem, recruiting technical developers who were sympathetic to user-centered design and 'forcing them to co-operate' "

Technical Unit System Manager: "freedom from performance deadlines to permit technical experimentation prior to producing system requirements and iterative design with "HCI-evaluation rather than case study evaluation"

Technical Developer(3): "starting the design cycle with a technical prototype, followed by user-interface evaluation, rather than starting with user-requirements investigation".

Individuals’ descriptions of the constraints of iterative, user-centered design appear to center on the need to manage problem definition and to define agreed design objectives, but these needs are interpreted in very different ways. The final contribution lies with technical developer(3), who had argued that defining system requirements should be the preserve of technical developers rather than of "flower arrangers". He commented that:

> "I find it incredibly difficult to understand why people find computers difficult to use."

DISCUSSION OF CASE STUDY FINDINGS

The main themes found in the case study findings are illustrated in a social-action model of design, in the form of a "rich picture" (Checkland, 1981); this is shown in Figure 4.

Two conceptual barriers to user-centered design were detected during the case study analysis. The first barrier was one of *visibility*, erected by the technical developers to resolve the conflict between the need to deliver technology in a short timescale while maintaining a work-status which depended upon perceptions of technical "expertise" and the need to engage in an intensive learning process. By reducing the visibility of their work the technical developers were able to conceal their learning process and to control others' perceptions of design progress. But once erected, this barrier had an additional benefit: it permitted the technical designers to monopolize the definition of system form. As the form of the technical system was no longer exposed to view, it could not be challenged and so the system scope could be redefined to reduce the perceived system complexity which followed from consideration of the context of use (scientific reductionism).

The second barrier was one of *perceived relevance:* the user-requirements evaluators had a culture where knowledge was formally recorded; information did not exist in the public domain unless it had been circulated in a written form, whereas the technical developers avoided written “documentation”: knowledge was communicated and validated informally. Technical developers were able to ignore the work of user-requirements evaluators by refusing to acknowledge written design representations as valid forms of information and by avoiding personal contact with the evaluators. When challenged by evaluators with a technical background, the technical developers avoided the threat by defining what technical knowledge was relevant to the project. People without "relevant" technical knowledge were excluded from effective decision-making: evaluators were not consulted about a fundamental decision on the nature of the proposed technology, while non-team members who were considered “qualified to comment” because of their technical backgrounds were consulted.

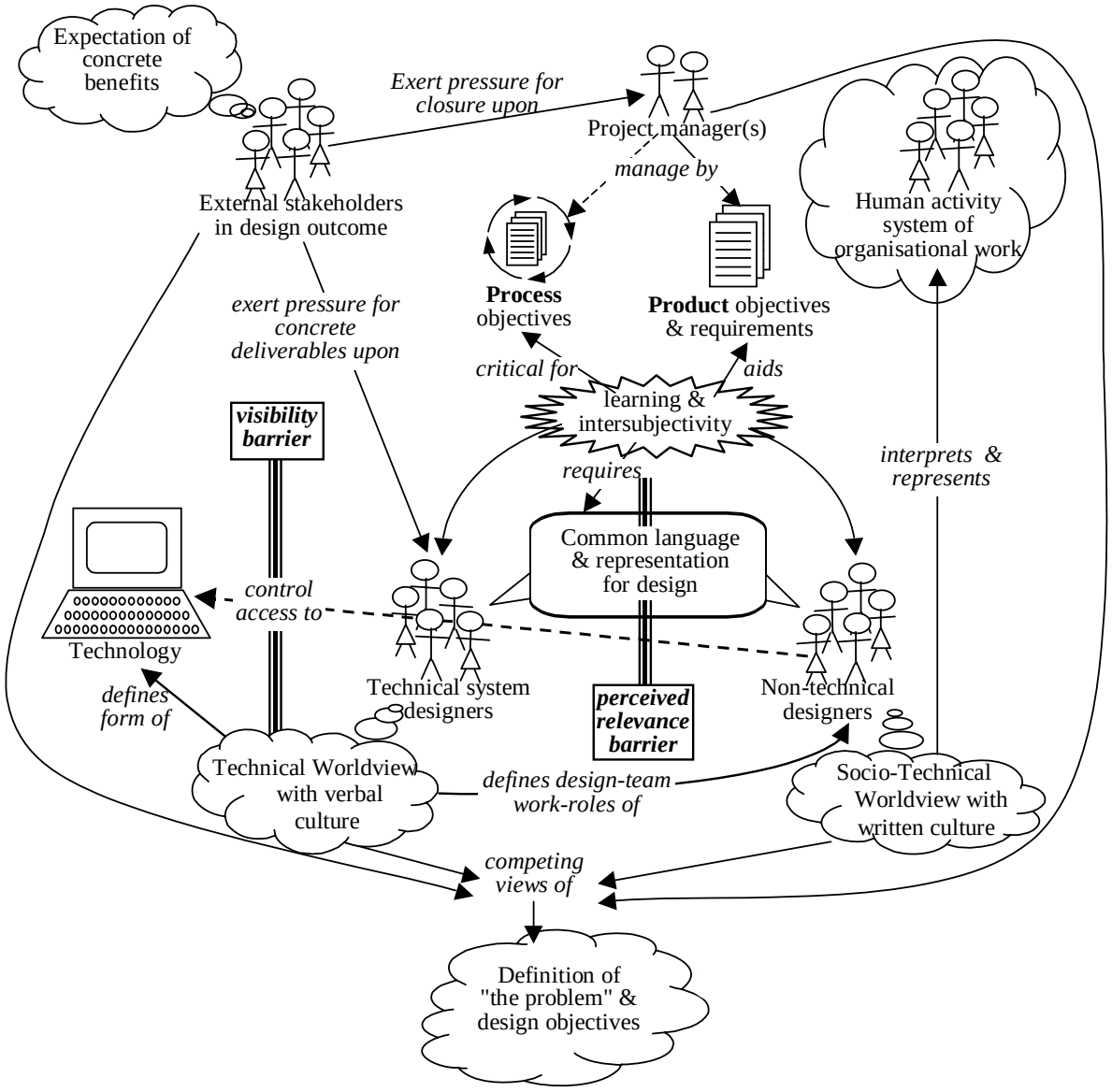

Figure 4: A Social Action Model Of Design Influences

Barriers To Visibility And Relevance: The Legitimacy of Domain Knowledge

Sociocultural Norms And Communities Of Practice

By controlling the visibility of technology and "relevant" knowledge, technical developers were able to manage perceptions of what could be achieved to such an extent that they could pursue other projects and interests, yet still appear to be overloaded with work. They felt no compunction in so doing because of differences in custom and practice between the two disciplines, which meant that they did not identify the psychologists as peer-group members (hence the pejorative term "flower arrangers"). Rosenbrock, (1981) argues that normative practice acts as a constraint upon social change in technical design, as human issues are excluded from technical designers' value systems. Lave & Wenger (1991) discuss the formation of normative practice by focussing upon the role of situated learning in “communities of practice”. To master knowledge and skill legitimately, individuals must adopt the sociocultural practices of a community of their peers, as in an apprenticeship. Identity, knowing and social membership are interrelated. For sociocultural values to

be adopted, the social group which constitutes the new community of practice must be cohesive. It was assumed that the use of a user-centered design method would ensure group cohesion; this proved not to be the case because the technical sub-group involved in the project defined their identity by their membership of a technical peer-group rather than by redefining their peer-group to include non-technical project team members.

The normative background of the technical developers formed their perception that a system design life-cycle started with the construction of a technical artifact. This affected their perception of the work-role of 'evaluator', which they defined as evaluating technical artifacts. The psychology background of the user-requirements evaluators, coupled with a perception that the design-cycle started with the investigation of potential contexts of use, meant that they were unable to contribute meaningfully to technical design discussions. The formal ways in which the user-requirements evaluators had been trained to communicate system requirements - producing written work analyses and task-definition documents - were viewed by the technical developers as valueless. Technical developers have a much less formal culture and are highly resistant to documenting the outcomes of their work until it is completed, so they did not see the dynamic requirements documents produced as a result of user-evaluation studies as relevant.

Sociocultural value-systems were also important in defining the *scope* of the design. System purpose, functions and use were defined differently by the two worldviews of technical developer and user-requirements evaluator. Technical developers were able to limit the scope of the system design to a conceptualization of *function*, excluding debate on system *purpose* and *use* as "irrelevant". The lack of impact experienced by the user-requirements evaluators constrained the range of activities which they were able to undertake, reinforcing their role as defined by the technical developers. Technical developers were also able to control how team activity was co-ordinated, as the redefined design tasks centered upon the availability of technical system prototypes.

Domain-specific learning is not seen as a legitimate activity in traditional (technical) system development: developers are hired for their existing expertise in particular application domains. Curtis et al. (1988) discuss the role of the "expert designer" in educating design-team members and stakeholders in both technical possibilities and domain-specific information system requirements. It is possible that, if the senior technical developer had seen the acquisition of domain-specific experience (in this case, experience of learning technology) as legitimate, the design team might have been more cohesive. As it was, neither peer-group had any expertise that the other could value, so they lacked a "common language" for design dialogues. The Project Director commented that he had underestimated the amount of learning about how to approach user-centered design which would be required by *all* of the project team, assuming that the use of a user-centered design method would provide sufficient basis for them to view project activities and outcomes in the same way that he did.

Technical developers appeared to be willing to sacrifice the main objective of the project - the ability of the system to support interactive learning - to maintain technical credibility. The professional identity of technical developers lies in their technical expertise: the ability to understand technology and to make decisions concerning appropriate forms of technology. When such identity was threatened by being required to explore unfamiliar forms of technology, it appears that their response was to maintain peer-group sociocultural norms at the expense of higher-level objectives.

The Management of Meaning

The management of meaning, in terms of powerful actors being able to define the work environment for others is a recognized phenomenon in the field of management leadership theory (Smirchich & Morgan, 1982). But the ability of technical developers to manage meaning for others is relatively unexplored in the information systems literature, except for a study by Markus & Bjorn-Andersen (1987), who analyze the power of systems developers over system users.

The technical developers were able to define the meaning of the term *evaluation*, and thus team members' work-roles, through their refusal to participate in design collaboration with non-technical actors. The evaluators' initial, rich work-role of investigating socio-technical, human-computer interaction requirements of a learning-support system was reduced to technical product evaluation. By redefining the role of user-requirements evaluators on the project, technical developers were able, deliberately, to shift the emphasis of the project from a primarily user-centered process, to a technology-centered process, by 'rotating' the design life-cycle by 90^{o} as shown in Figure 5. This process is more similar to a "traditional" system development life-cycle than a user-centered life-cycle: while it reduced uncertainty for the technical developers, use of this life-cycle model subverted the possible outcomes of the project from a fairly early stage.

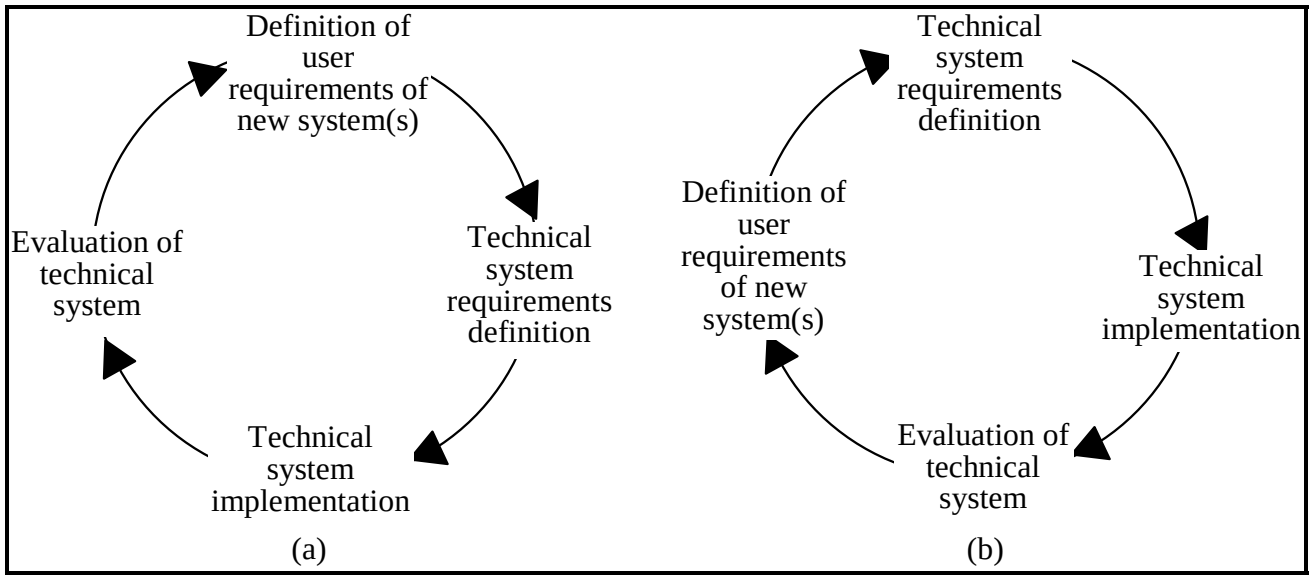

Figure 5: Comparison of (a) user-centered system development life-cycle with

(b) traditional (technology-centered) system development life-cycle

The term *design* was also appropriated, to represent technical design. It became clear that the technical developers did not consider user-requirements evaluation as design work; they used the term design to refer exclusively to the design of technology. The term *system*, used in the sense of a socio-technical system by the psychologists, was always used by technical developers to refer exclusively to the technical system. The conceptual power which technical developers were able to exert over the psychologists (including the Project Director), in defining "appropriate" technology and in providing technical prototypes, permitted them to control notions of relevance and legitimacy. The meaning of "design" was subverted over time until all members of the project used it to refer to the technical design and referred to investigation of the social system as "user-interface analysis".

The subjugation of social system design to user-interface analysis permitted another subversion of the user-centered design philosophy. Bjorn-Andersen (1989) comments that human-computer interaction is often far too narrowly conceived in information system development: interaction means much more than screens, buttons and function keys. Human-computer interaction means designing the whole computer system to support work tasks in an appropriate and user-centered manner, allowing maximum support and autonomy to the system user (Preece et al., 1994). By reducing socio-technical system design to user-interface design, the technical developers were able to manage uncertainty by reducing the scope of the technical design. They no longer had to cope with the complexity of

alternative system purposes and uses, except in considering appropriate keystroke combinations for tasks which they themselves had defined.

The desire to experiment with new technology apparently drove much of the technical design, to the exclusion of "the user-interface". While building two prototypes and throwing one away may be appropriate for *experimental* prototyping, it is far from appropriate for *evolutionary*, user-centered prototyping (Floyd,1984). It appears that the technical developers were not interested in user perceptions of the system at all and did not see the first prototype as intended for user-requirements elicitation, even though this was the objective which they were given by the Project Director. The project had been constituted to investigate how technology might support innovative contexts of learning. By making the context of use follow technical design, rather than technology following the requirements of new contexts of use, the system design ceased to be either innovative or usable.

Managing Task Interdependencies

The management of uncertainty is a recurring theme in this study. By refusing to interact with other team functions, technical developers could determine the scope of the system design. When faced with emergent or unclear design objectives, they reduced design collaboration until they had defined their own system objectives through technical experimentation. In response to the efforts of the Project Director and the evaluators to exert structural power by defining collaborative tasks with imposed deliverables and deadlines, the technical developers gained control over the process and *redefined the nature of the process* by using structural dependencies between the tasks. The technical nature of the production of prototypes for evaluation gave technical developers the ability to define what the system would do, as the psychologists did not have the expertise to influence detailed discussions of technical system function. Although the psychologists tried to experiment with the technical system so that they could participate in technical design decisions, this appears to have been thwarted by their dependence upon the technical developers to configure the technology. Both peer-groups separated their own work from that of the other group, to ensure manageable, separate tasks. In so doing, each discipline made their work-outcomes irrelevant to the other.

CONCLUSIONS

The employment of a user-centered of system design method was critical for this type of research project, as ease of learning, ease of use and exploration of system purpose in supporting interactive learning were of major importance. But it was observed that the use of a user-centered method was insufficient to produce an appropriate system outcome. There were two critical management processes which determined the effectiveness of the user-centered design method: defining the starting-point of the system design life-cycle and managing expectations of work roles and tasks.

The emphasis which different members of the team placed on social and on technical aspects of the design depended upon their worldview (Checkland, 1981); this was largely defined by their identification with a professional peer-group rather than the formalisms of the user-centered design method. The meanings attached to various forms of design knowledge and the technology-centered definition of "relevant" knowledge were central to the failure of this project.

A critical problem which arose was the need to plan for and legitimize domain-specific learning. The dynamic nature of knowledge concerning the social system in this project echoes the work of Galliers & Swan (1997), who concluded that effective design must be able to communicate and represent *informal* knowledge about a design, in a way which does not ‘concretize’ such knowledge, but makes it open to debate and change. This study demonstrated that the form in which such knowledge is presented is critical, as preexisting sociocultural expectations must be managed. A major area of concern was the lack of a common language or representation for collaboration between peer-groups from different disciplines.

Group design is a research domain which has not been explored in the literature to any great extent. A core issue appears to be the tension between individual problem conceptualization and cross-disciplinary negotiation of design scope. Peer-group definition and the values and meanings attached to different work-roles and tasks require further investigation, with respect to issues of legitimacy, relevance, visibility and uncertainty. These are issues which are not specific to user-centered design groups, but which pertain to most design projects. IS development managers must address issues more fundamental than the employment of a particular methodological approach, if their organizational information systems are to be truly user-centered..

REFERENCES

Amoako-Gyampah, K. and White, K.B. (1993). User Involvement and User Satisfaction. Information and Management. *25*, 1-10.

Avison, D.E. and Wood-Harper, A.T. (1990). Multiview: An Exploration In Information Systems Development. Blackwell Scientific Publications. Oxford

Barki, H. & Hartwick, J. (1994). Measuring User Participation, User Involvement and User Attitude. MIS Quarterly, 59-82

Baroudi, J.J., Olson, M.H. and Ives, B. (1986). An Empirical Study of the Impact of User Involvement on System Usage and Information Satisfaction. Communications of the ACM. *29*(3), 232-238.

Beath, C.M. & Orlikowski, W. (1994). The Contradictory Nature of Systems Development Methodologies: Deconstructing the IS-User Relationship in Information Engineering. Information Systems Research. *5*(4), 350-377.

Bjorn-Andersen, N. (1989). Are Human Factors Human? in H.K. Klein & K. Kumar (Eds.) Systems Development For Human Progress. Elsevier Science Publishers. .North Holland.

Cavaye, A.L.M. (1995). User Participation In System Development Revisited. Information & Management. *28*, 311-323

Checkland, P. (1981). Systems Thinking, Systems Practice. John Wiley & Sons. Chichester.

Curtis, B., Krasner, H. and Iscoe, N. (1988). A Field Study Of The Software Design Process For Large Systems. Communications of the ACM. *31*(11), 1268-1287.

Flanagan, J.C. (1957). The Critical Incident Technique. Psychological Bulletin. *1*, 327-358

Floyd, C. (1984). A Systematic Look At Prototyping. in R.Budde, K.Kuhlenkamp, L.Mathiassen, & H.Zullighoven (Eds.) Approaches To Prototyping. Springer-Verlag Books

Galliers, R.D. & Swan, J. (1997). Against Structured Approaches: Information Requirements Analysis as a Socially Mediated Process. in J.F. Nunamaker, Jr., & R.H. Sprague, Jr., (Eds.) Proceedings of the Thirtieth Hawaii International Conference on System Sciences. Information Systems Track. IEEE Computer Society Press. Los Alamitos. CA, 179-187.

Giddens, A. (1993). New Rules of Sociological Method. (Second Edition) Polity Press. Cambridge. UK

Glaser, B.G. & Strauss, A.L. (1967). The Discovery of Grounded Theory. Aldine Publishing Company. NY

Kappelman, L.A. & McLean, E.R. (1992). Promoting Information System Success: The Respective Roles of User Participation and User Involvement. Journal of Information Technology Management. *3*(1), 1-12.

Lave, J. & Wenger, E. (1991). Situated Learning: Legitimate Peripheral Participation. Cambridge University Press. Cambridge. UK

Lowe, A. (1996). An Explanation Of Grounded Theory. Working Paper, Dept. Of Marketing. University of Strathclyde. UK

Markus, M.L. & Bjorn-Andersen, N. (1987). Power over users: its exercise by system professionals. Communications of the ACM. *30*(6), 498-504.

Mumford, E. (1983) Designing Participatively. Manchester Business School. UK

Newman, M. & Robey, D. (1992). A Social Process Model of User-Analyst Relationships. MIS Quarterly. *16*(2), 249-266.

Preece, J., Rogers, Y., Sharp, J., Benyon, D., Holland, S. & Carey, T. (1994). Human-Computer Interaction. Addison-Wesley. Wokingham. UK

Smircich, L. & Morgan, G. (1982). Leadership: The management of meaning. Journal of Applied Behavioural Science. *18*(3), 257-273

Sumner, M. & Sitek, J. (1986) Are Structured Methods for Systems Analysis and Design Being Used?. Journal of Systems Management. *37*(6), 18-23.

Walsham, G. (1993). Interpreting Information Systems In Organizations. John Wiley & Sons. Chichester. UK

Yin, R. K. (1994). Case Study Research: Design and Methods. Sage Publications (2nd ed.). Thousand Oaks. CA