BEHAVIOUR & INFORMATION TECHNOLOGY, 1996, VOL. 15, NO. 4, 237–249

Software development: knowledge-intensive work organizations

CHRIS W. CLEGG, PATRICK E. WATERSON and CAROLYN M. AXTELL

Institute of Work Psychology, University of Sheffield, Sheffield S10 2TN, UK

Abstract. We report the findings from three studies of software development projects using a series of questions framed to provide a more detailed understanding than usually pertains of the management and organization, outcomes and derivations of work organization. We discuss some practical and theoretical implications of this work; in particular we conclude that these are knowledge intensive work organizations, that current theory is ill-equipped to address these practices, and that their analysis and understanding requires both organizational and cognitive explanations.

1. Introduction

One aim of this paper is to develop a better understanding of some software development practices through the analysis of three different case studies, using psychological constructs and approaches. Another aim is to use the field of software development as a vehicle for exploring some of the potential interdependencies that exist between organizational and cognitive psychological ideas and ways of thinking. In particular we are concerned with the issue of work organization, a major field of enquiry within organizational psychology, though largely neglected within cognitive studies. The specific objectives of this paper are:

- to describe three case studies of software development;
- to identify some existing gaps in work organization theorizing, and reveal how the inclusion of some cognitive thinking would improve our understanding;
- to speculate on some implications arising from this work.

The paper is organized in four further parts. First, we provide a brief review of some relevant literature. Then we introduce the case studies, outlining the context, objectives and research methods of each study. Next we summarize the key findings using a standard set of questions. Finally we speculate on some of the theoretical and practical implications of this work.

2. Review of the field of work organization

The aim here is to draw attention to some major trends in the literature that help place our own work in context. We wish to make six major points. First, we highlight the historical differentiation that exists between organizational and cognitive psychological studies. This takes several forms, and includes separation by issue, methods and underlying paradigm. (For a fuller description of this argument, see Clegg (1994)). We believe such fragmentation is more a function of the way these two psychological sub-disciplines have developed than of the intrinsic nature of the issues they address. Second, it is apparent that little interest has been shown by cognitive psychologists in the practice of cognition in work organizations (Eysenck 1990, Eysenck and Keane 1990). Third, there has been an enormous amount of work by organizational psychologists (and those in the field of organizational behaviour) studying the phenomenon of work organization and job design (Parker and Wall 1995, Wall and Jackson 1995).

Fourth, we draw attention to some of the major difficulties with approaches to the field of work organization, as currently conceived by organizational psychologists. The dominant theoretical and empirical traditions are socio-technical systems theory (Cherns 1976, 1987, van Eijnatten 1993) and the job characteristics approach (Hackman and Oldham 1976, Karasek 1979). Much of this work has focused on blue collar industrial workers in relatively constrained jobs and has explored the impact of certain types of jobs on the well-being and performance of the job holders. The literature tends not to consider why job designs take the form they do. For example, relatively little work in this area investigates why it is that some jobs are simplified and deskilled. Nor are there many studies of how work organizations actually operate in practice, focusing on what individuals do in some detail, who they interact with and how their work fits in with those around them. Furthermore, there has been little in this field examining

new forms of work organization, for example involving knowledge workers, project teams, temporary organizations (Parker and Wall 1995). We conclude that empirical and theoretical work in this field has been unnecessarily restricted in its scope and coverage. Our own work attempts to address some of these gaps.

Our fifth point is that such generalizations do not apply to all work in this field. For example, action theory, the dominant approach amongst German industrial and organizational psychologists, embraces much more of a cognitive emphasis and has considered the detail of how jobs actually work in practice. Nevertheless some of the criticisms above apply to those utilizing this framework. In particular, action theoretical work has not addressed how work organizations arise. Furthermore, such approaches tend to focus on the individual, ignoring the social and distributed aspects of work organization (Frese and Zapf 1994, Hacker 1985). Sixth, there is an emerging literature on the cognitive aspects of working, and the field of distributed cognition offers potential for exploring the overlap between cognitive and organizational approaches within this domain (Hutchins 1989, 1991). Some of this looks in much more detail at the way work operates, about detailed working practices. Furthermore some work in this field has been undertaken in software development (Curtis 1988, Flor and Hutchins 1992, Olson and Olson 1991).

Recognizing the trends identified above, our own work can be located primarily in the socio-technical tradition. Our studies attempt a fuller understanding of work organization practice, focusing on detailed descriptions of the organization of software development project teams, their outcomes, and their derivations.

3. Case studies

The ideas described in this paper were generated through undertaking three case studies. They are concerned with software development but each has a different emphasis. The first case describes the implementation of a structured method (Information Engineering) and its associated CASE (Computer Aided Software Engineering) tools in a large private company; they were for use by software developers designing new software systems for internal company use. The second case was undertaken in the same company; in this instance we studied one particular software development project. The developers working on this project became users of the CASE tools described in the first case. The third study was carried out in a large organization in the public sector. Again we studied a large-scale software development project: our principal role was to investigate a new method of involving end users in software development, a social rather than technical innovation.

Whilst the cases focus on some quite different activities, the organizations share some characteristics. They are large and undertake in-house development of software for use by their own employees. They are also at the forefront of developments in the field of software development. Thus the company in cases 1 and 2 is technically advanced; that in case 3 is technically strong, but also advanced in its thinking concerning the role of end-users. We stress that we sought access to these organizations because of their innovative practical work in this field and to help us explore and develop the ideas discussed in this paper, rather than as a means of drawing representative conclusions about the nature of software development more generally. We make no claim that these organizations are representative of others; in our experience it tends to be relatively large, successful organizations that undertake innovations of the kind described here, and also that allow access for research. These circumstances are clearly different to those of many other companies, including those developing software as their *raison d'etre*.

The case studies involved intensive fieldwork, working towards a set of objectives agreed with each organization. In each instance our role was to attempt to understand a particular practical situation and to make recommendations for change and improvement, working within the tradition of action research (Foster 1972). Our research team comprised a mix of organizational and cognitive psychologists.

We report the findings from the cases against five principal questions. The questions were framed in an attempt to provide a more detailed understanding than usually pertains of the management and organization, outcomes and derivations of work organization, to overcome some of the deficiencies identified in the review above:

- How is the work organized and managed?
- With what effects?
- How did the work organization arise?
- What disrupts the pattern of work organization?
- With what effects?

The criteria and measures used in each study were common in some instances and varied in others depending on the context and objectives of each study.

4. Case study 1: Implementation of IE and CASE

4.1. *Context and objectives*

This study was undertaken between October 1992 and February 1994 in the Information Technology Development (ITD) department of a large national financial services company based in the UK. This department was responsible for planning, developing, implementing and supporting major IT projects for use within the company. The department employed around 250 people and was highly

regarded for developing advanced systems using the latest techniques. One section of the department was responsible for examining the applicability, and undertaking the implementation of advanced technologies for use within the company generally and within ITD. As a result of their work, the company decided to introduce for use within ITD a structured methodology, Information Engineering (IE), and its associated Computer Aided Software Engineering (CASE) tools. These can be regarded as state-of-the-art computer-based methods and tools for supporting software development throughout the life cycle, from strategy to maintenance and support. The principal objectives of this investment were: to improve the quality of deliverables (new systems); to increase the productivity of developers; and to provide for greater efficiency in ITD.

The objectives of our study, agreed with the company, were to examine and help optimize the impact of Information Engineering and associated CASE tools within the company from an organizational (rather than technical) perspective.

4.2. *Research methods*

We used a number of research methods to address the objectives and questions identified above. These included undertaking semi-structured interviews with all six members of the implementation team, their senior manager, and 32 others from different levels and functions including IT planners, developers and senior managers. Some of these individuals were interviewed several times. We also administered a questionnaire survey to all relevant developers in ITD, receiving 74 completed questionnaires (a response rate of 54%). (For further details see Waterson *et al.* (1996); as in the other studies, copies of research instruments are available on request from the authors.)

5. Case study 2: Charging project

5.1. *Context and objectives*

This study was undertaken in the same company as described above. We again worked within ITD, this time with a project team developing an accounting system for use in the company's Finance Department. Part way through this project, the team became users of the CASE tools whose implementation was described above. This study took place between October 1992 and May 1994. The new accounting system was for charging external customers for a wide range of financial services (hereafter called the Charging project). The system comprised over 1 million lines of code. A computer-based system for charging existed, so part of the task involved updating old software and ensuring the old and the new were compatible. Thirty people were involved on a full-time basis on this project. During our study the new system was in detailed design and coding stages.

The agreed objectives were to investigate the development process of the Charging project from an organizational perspective, and to examine the impact of the introduction of CASE tools on the project.

5.2. *Methods of study*

We conducted interviews with 24 individuals working on the Charging project, including the project manager, project leader, team leaders of the different sections, the analysts and programmers, senior managers, user acceptance testers, and experts in databases and CASE tools. Some key individuals were interviewed several times. (See Waterson *et al.* (1996); copies of research instruments available on request.)

6. Case study 3: Implementation and operation of user-centred development

6.1. *Context and objectives*

This study was undertaken in a large public sector service organization between February and June 1994. The study took place within the Information Technology department where we studied the development of a new software system for use by thousands of clerks and managers in offices all over the UK. A new form of user-centred development was being implemented at this time, explicitly aiming to foster end-user participation during the development of the above software. Our focus was less on the particular software, more on the new form of user-centred design which was being used for the first time. This case thereby is concerned with a social (as opposed to technical) innovation.

The agreed objectives of this study were to evaluate the new user-centred method as a process of system development in its context of use, and to give recommendations for improvements to the method.

6.2. *Methods of study*

Data were collected from a range of sources including users from offices in different parts of the country, system developers, and managers and other stakeholders from three different departments, namely Information Technology, Human Factors, and Network Services. Data were collected using a range of different research methods, including semi-structured interviews (of 21 users, 4 developers and 12 managers), observations and videos (of a group of 4 users for a 3 week period), questionnaires (of 67 users and 6

developers); and a tracer study using company documentation (following over 500 separate documents). (See Axtell *et al.* (1995); copies of research instruments available on request.)

7. Summary of findings

In this section we summarize the key findings from the case studies using the questions described earlier.

7.1. *How is the work organized and managed?*

In Case 1 a Project Manager was appointed to implement IE and CASE, reporting to the Planning and Infrastructure Manager within ITD. The implementation was organized in two phases, covering different parts of ITD. Our work was conducted during phase 1 of the roll-out. The Project Manager had five others in his team. Two of these held responsibility for the implementation in different parts of ITD, each of which represented a different part of the business. In turn these individuals also had someone reporting directly to them. There was, however, considerable flexibility in the allocation of work amongst the team and this was largely self-managed. An assistant provided clerical and secretarial support for the whole team.

The implementation team was responsible for: delivering hardware and software to developers according to an agreed plan; providing support to those developers when they used the tools (e.g. answering their queries); planning (but not delivering) training for users; and evaluating relevant software products that became available on the market. The team was project managed according to timescales and budgets. In particular they worked hard to deliver hardware and software according to the plans agreed with the developers.

All of the team were from technical backgrounds. Their expertise primarily concerned analysis, programming, system support, CASE tools and technical project management. Common perceptions were held of the role of users by the implementation team. Users received a standard tool to enable them to fulfill their role requirements, e.g. a tool to undertake analysis. Each user received detailed training in their own tool, and awareness training in the immediately preceding and following tools in the life cycle. Their main role had been to help the IE/CASE team identify requirements. A later role for users concerned negotiating an implementation schedule. To the implementers, user participation primarily meant involving the users in getting the requirements right in the first instance, and thereafter in agreeing a schedule for implementation.

This was primarily a technology implementation project. The human and organizational factors were under-represented in the implementation process. No one was given the role of, or invested time and effort in, adopting a more socio-technical perspective. There was no consideration of the likely impact of the new methods and tools on how the work of the developers would and could be organized. No one sought to simplify or business process re-engineer (BPR) their work organization. Whilst the Project Manager and his boss recognized that BPR would be appropriate, this was viewed as something to worry about later. For now, IE and CASE represented a change in the medium of software development rather than a change in working practices. Nor were other possible perspectives on change represented within the implementation team; no one advocated user-centred design, nor laid stress on this as an educational change.

In Case 2 two factors helped shape the organization of the project. Because of its strategic importance, senior managers decided to contract out the writing of new software to an independent software house. Furthermore, an existing computer-based system was in operation for charging customers, so part of the task involved conversion of existing code to match the requirements of the new system and interface with it. The Charging project was managed by a Project Manager assisted by a Project Leader. The project team was organized in three groups, each with a group leader. The three groups held separate responsibility for: resolving queries raised by the external contractor; converting old into new code; and user acceptance testing.

The Project Manager and one of the group leaders had been responsible for selecting people to join each of these teams. In practice the division of labour was more flexible than this description suggests. Wide fluctuations in workload were handled in three ways. In the first instance, people in the group were simply expected to work harder. Second, staff could be transferred from one group to another within the project team. This happened fairly frequently. And third, extra staff could be drafted in from elsewhere in ITD. This happened rarely. In practice the team and the groups within it were largely self-managing. They were expected to meet various targets and goals but the Project Manager did not specify how this should be done.

Five principal domains of expertise were incorporated within the project team. These were: project management expertise; knowledge and experience of software engineering techniques and tools (such as SSADM); computational knowledge of the new application (concerning hardware and software); detailed knowledge concerning the application (e.g. regarding its inputs and outputs); and knowledge of the domain in which the application will be used (e.g. concerning the task of charging customers).

Approximately 30 people were working within the three groups in the Project team and all the above categories of knowledge and expertise were represented. We draw attention to knowledge of the application domain. A

number of ex-users from the Finance department were recruited onto the project team but two comments should be made about their participation. First, they were no longer users—they expected to stay in ITD. And second, their role was focused on User Acceptance Testing which involved them in testing pieces of code using acceptance protocols. Their role should not be construed as enabling user participation in the development of the system. Rather, the principal roles for users, as in Case 1 above, had been to help the analysts derive a set of requirements early in the development process (through the use of SSADM), and, during the current phase, to test and accept the new software.

The three groups worked together in an open plan office. Patterns of communication were open and informal. Nevertheless the leaders responsible for each group took primary responsibility for ensuring that necessary information was exchanged; they were key figures acting as boundary spanners in this process. This was important because the work across the three groups was highly interdependent. For example, the queries group needed to know what problems were occurring with user acceptance and also with the conversion of old to new code. The group leader responsible for handling the queries raised by the external contractor proved to be an important figure. In part this was a function of his boundary spanning roles, both between the three groups and the two companies. Also, however, he had been heavily involved in the original analysis work with the users in Finance. He was the only individual holding knowledge across all five categories identified above; his knowledge was critical to this project.

A striking feature of this work organization was that it incorporated a mix of formal and emergent design. The three different groups were instituted and appointed formally but how they operated, communicated and interrelated was left unspecified and emerged over time. This dynamism and flexibility proved highly effective for sharing information, for problem solving and for joint learning. Negotiation and mutual adjustment were characteristics of this work organization.

At the time of our study the system in Case 3 was nearing completion and a formal regional pilot test was about to take place. Here we draw a distinction between the original method of working and that studied during our research. We describe the original method first. Four different groups of people were involved in the development process. These were: a group of software developers from the IT department who, at this stage, were responsible for completing the programming, testing the outcomes, and delivering the new system to meet a set of requirements; groups of users from local offices who were responsible primarily for helping improve the usability, and in some cases the functionality, of the new system; a small group of people from the Network Services department who were responsible for organizing and managing the user side of the new method of user participation and the necessary liaison with developers; and a group of Human Factors specialists in the organization who advised on the design of the new software system (e.g. concerning its usability) and the new method of user participation. These groups had different reporting lines and accountabilities, and there was no single manager responsible for the new method of involving users.

In the early days of this method, groups of users (of around 6–8 people) were seconded from local offices by Network Services to visit and work with the development team for up to five weeks at a time. The users were from different types of offices and held different jobs; Network Services were responsible for ensuring that the desired mix of users was represented at some stage during development. During their stay users were involved in two primary types of work. The first involved working individually with a developer (called here a Cooperative), jointly developing a particular part of the software. The user was expected to help ensure that the design of the software was usable, though functionality issues also arose. In the second role users worked together as a small group of 3–4 people (called here a User Group) evaluating new bits of software, for example by undertaking realistic role plays and scenarios designed by themselves or by people from Network Services. During their secondment a user would be expected to spend some time working in a Cooperative and some time in a User Group, the proportions varying depending on the stage of development and the length of their secondment. This user-centred method had been in operation for approximately 6 months prior to our involvement in the project. Eighty seven users had taken part in the process. However the method of user participation operated differently to that described above whilst we were involved.

During our involvement the developers were under intense pressure to complete the system to meet the pilot deadline. As such all Cooperatives were suspended by the manager of the developers. Users continued to be actively involved but only in User Groups. We studied a number of such groups. In practice they undertook a number of activities. These included: helping develop realistic task-based scenarios (based on their local office experience) which could be used to test new software; inputting data into the data base so that any trial tasks would have data on which to operate; undertaking the scenarios to test the usability and functionality of the new software; designing screens for an on-line help system; general 'trouble-shooting' work including establishing the priorities of problems emerging; and usability testing, for example commenting on the interpretability of error messages.

During this period the User Groups were given work by staff from Network Services, reflecting their views of priorities at the time; these were influenced by the manager of the software development team, in part influenced by the

needs of the developers. Because of the lack of direct contact between users and developers, the outputs from the User Groups went back to Network Services staff who made decisions about how to act upon them. The work organization within the User Groups was informal and flexible. Users were told what to do but not directed in how to do it. They had considerable autonomy in their work.

In each of the three cases the work was organized in small, temporary project teams with a hierarchical structure and some lateral division of labour. The work organizations operated both planfully and emergently. There is evidence in each case of flexible working, of mutual adjustment and of negotiation. The informality and flexibility of work organization is ideal for mutual learning and for the process of education within the team. In each instance the work is 'minimally specified'; people are required to meet certain targets or requirements but are not told how to do it. This 'looseness' in specification gives scope for flexibility and learning in the face of uncertainty and complexity, but, of course, can also result in apparent inefficiencies.

We believe that knowledge and expertise are central characteristics of the design and operation of these work systems. They are knowledge-intensive work organizations. Multiple forms of expertise were included in each of the project teams although this varied case by case, particularly concerning the scope of involvement of end-users and their domain knowledge. Such work organizations can only be as effective as the knowledge and expertise incorporated within them. In two of our case studies, these were almost exclusively technical knowledge systems. The third was more plural, incorporating knowledge of the user domain, but some of this was eventually marginalized in the drive to meet requirements laid down by senior managers.

7.2. *With what effects?*

Evidence from the survey undertaken in Case 1 reveals mixed expectations regarding the likelihood that IE and CASE would meet their objectives. Greatest confidence was expressed regarding the increase in the skills set of developers, the increase in the productivity of developers, and the improved quality of deliverables. Least confidence was expressed over whether or not IE/CASE will lead to more flexibility in staff allocation, more effective prioritization of work, better project management, improved job satisfaction and improved communication between users and developers. Regarding their views of the implementation process, the developers were most positive concerning the adequacy of training and of the way the introduction was being managed. They were least positive about their opportunities for participation and the plans for re-engineering the development process.

But these overall results mask some important systematic differences within the sample of survey respondents. In general, the CASE implementation team were the most positive about whether or not IE/CASE will meet the required objectives. The overall pattern is that those involved earlier in the life cycle and/or higher up the organizational hierarchy, tended to be more positive about the likely impacts of IE/CASE. The most negative were the users working later in the life cycle. Less than 50% of the programmers believed IE/CASE will meet the principal objectives, namely improvements in productivity, efficiency and quality. Feelings of ownership of the new methods and tools reflect a similar pattern and overall were low, particularly for the main users, the analysts and programmers.

Our interpretation is that this implementation provides an example of technology-led change (Clegg *et al.* 1994). The vast majority of organization resources were invested in technical issues, to the relative exclusion of organizational (or joint socio-technical) concerns. The implementation team worked hard and successfully to ensure a well managed project implementation (Blackler and Brown 1986). This had its successes, most obviously that the technology was implemented and used. But it also had its difficulties. The technology was treated as a change in medium and thereby much of the potential for benefit lost. It seems likely that local efficiencies will result rather than fundamental improvements in overall productivity. There was also considerable scepticism on the part of users as to whether or not the investment would realize its objectives, and at the same time low feelings of ownership of their new technologies. This is particularly so for the major categories of users, analysts and programmers. In the next study we report in more detail the experiences of a sample of the CASE tool users.

In Case 2 the way the work was organized proved to be highly effective in so far as it allowed flexibility, mutual adjustment and learning from one another whilst on the job. This design of work is consistent with two core sets of ideas. First, very complex work processes cannot be entirely routinized and there should be scope for informal negotiation and mutual adjustment (Mintzberg 1979). And second, the socio-technical principle of minimal critical specification pertains here; in this view the work groups should know what is required of them, but how they do it should remain within their own powers of organizing and structuring.

On the debit side there were some problems with how the work was organized. There were substantial disadvantages in 'outsourcing' development to a separate software house. It meant that there were substantial communication, understanding and learning blocks between the two groups of people working on the same system. The very informality, closeness and mutual adjustment that proved such a strength internally, of course was impossible to achieve with people elsewhere. Furthermore, the external developers had a commercial contract to deliver code to achieve certain

functionality. The internal developers were very concerned that they would be left with the problem of supporting and maintaining whole parts of the application that they did not understand, had not written, and was not written with subsequent maintainability as a key criterion.

We conclude that this project was also technology-led in that the vast majority of organizational resources were concerned with technical, rather than social concerns. Most of the expertise in the project team was of a technical kind, and the ex-users employed on the project team were acceptance testers whose role was to test the functionality of the new system. As in Case 1, no efforts were made to adopt a socio-technical approach to designing systems for charging customers, nor to re-engineering and simplifying the processes of charging customers.

Turning to Case 3, the managers were positive about the new method of user involvement and strongly committed to the principle of user participation. There was a shared assumption that involving users helps make for 'better' systems and that they have a greater role than simply improving usability. Furthermore, in contrast to Case 1 above, those closest to the new method of working were most positive and committed to it.

Users were positive about the new method of participation and their expectations of, and commitment to, it were high. Only a few expressed reservations and most of these were concerned with cessation of the Cooperatives. Most of the users who had been involved felt they had 'quite a lot' to a 'great deal' of influence over the new software, especially when they had worked in a Cooperative. A further criticism, especially during the later parts of development, concerned the lack of communication between users and developers, and the perception that the two groups remained quite separate. Some users who had returned for a further secondment with the development team reported that they felt more confident, having a better idea of what was expected of them the second time around. Many users wished to be involved in the process again. The User Groups were seen as worthwhile and interesting by the users, and they spent a great deal of time actively helping design the new system; their role was concerned with much more than usability. Furthermore, reports from users in the pilot site confirm that users regard the new system as a vast improvement on the old.

For their part the developers were also generally positive about the new method of participation. They agreed with its goals and felt it was a significant advance upon earlier efforts in this area. The majority of developers took the view that the method was largely successful in meeting its objectives. All agreed that the method could be improved with better preparation and training, with more realistic deadlines and with better management. Commitment to the method was 'moderate' to 'high'. The best aspect of the method was seen as being able to get quick answers to questions about business requirements whilst enabling the developers to concentrate on what they knew best, i.e. the technical matters. The worst aspect of the method concerned the time taken up working directly with users. For example, Cooperatives were sometimes seen as too open-ended and could be an inefficient use of their time, especially at the beginning, when for example, they were asked to train users on basic computing skills.

Developers were also critical of coordination between different parts of the system. This was particularly so during the period of our investigation, and was borne out by the results of a tracer study in which we found that many of the users' reports and finding did not make their way back into the development process. The cycles of development and evaluation work became asynchronous and inefficient.

Overall, our interpretation is that these work organizations can cope effectively with variations in internal and external demands. Their size, informality and mix of planning and opportunism (Broadbent 1993) appears to provide a good environment for mutual adjustment and learning. These seem effective organizations for handling complexity and uncertainty. But such organizations are limited by their inclusions and their exclusions. Cases 1 and 2 above, represent very partial and incomplete knowledge systems, most obviously in their almost exclusive incorporation of, and attention to, technical issues. Even in Case 3, which constituted a serious attempt to include user knowledge and expertise, the method broke down under severe pressure to meet deadlines. Under pressure, it was the user knowledge which was excluded.

We believe these work organizations proved successful at avoiding messy and chaotic implementations. In each the new systems were implemented and they did function. But we draw attention to the relatively poor levels of performance evident in these case studies. In Case 1, many of the end-users of IE/CASE were sceptical that the implementation will meet its objectives. They report low feelings of ownership. Through the focus on technology, the organization is missing a major opportunity to reorganize how it undertakes system development. Local efficiencies may accrue, but genuine improvements in productivity and quality will be unlikely without a more organizational perspective. Evidence from Case 2 supports this argument. Again the change is technology-led. According to key informants the failure to simplify or re-engineer the application domain meant that the new system took 10 times longer to develop than was planned. And finally in Case 3, a number of senior management decisions led to the severe dislocation and partial suspension of what had been a successful system of user participation. A further problem in this organization concerned the lack of integration between different aspects of the development process. Overall, we conclude that these systems were not meeting their objectives and much of their potential was not met.

7.3. *How did the work organization arise?*

The implementation in Case 1 was seen as a technical project requiring careful project management. We believe it was like this because this was how it was seen by the key actors, the Project Manager and his boss, the Manager of Infrastructure and Planning. They determined the size of the team, the scope of their work, the skill set required to meet their objectives, and they recruited the people to join the team and undertake the implementation. They recruited people with technical expertise. Our interpretation is that their choices reflected their representations and views of the demands and requirements of the tasks facing the implementation team.

Interestingly in our discussions and interviews with members of the project team, we found a hierarchical dimension to the representations of the implementation which they expressed. The most junior members of the team, who were largely involved in physically implementing and maintaining the tools, emphasized the technical aspects of their work. The two individuals responsible for the different areas of ITD also emphasized these aspects but added a project management dimension to the work. The Project Manager stressed both these components but also recognized the political aspects of his work. However, these political issues were concerned with getting the new technology authorized and implemented, not with changing organizational structures and processes.

We offer two further speculations. First, we believe that two factors helped shape the views and actions of these key actors, namely their professional training in IT, and their company's normal style of managing implementations of this kind. In fact both these individuals had extensive experience in other companies in IT and in engineering more generally; this suggests to us that their professional training and education were highly significant. And second, it seems likely that work organizations of this kind become self-perpetuating. Managers operate within a social context in their companies and their actions are subject to the influences and expectations of those around them, their superordinates and subordinates. They recruit into their project teams people who have what they regard as the appropriate forms of expertise, knowledge and skill. These people themselves are likely to have similar backgrounds and to have been trained and educated similarly. Over time, the views of all these actors may become substantiated and confirmed by the people they report to and work with. Furthermore, it is inevitable that the roles into which people are recruited are significant influences on their sphere of operations, what they do, their actions. These actions in turn influence their views of their role and of the enterprise in which they are engaged. Together, these forces shape the cognitions of those taking part in the implementation and provide a means by which actions and decisions may be made and justified. In this case, the exclusive technical background of management and implementers reinforce, as well as perpetuate, the rationale behind a technology-led approach.

In Case 2 the decision by senior managers within the company to contract out writing of the new software meant that handling queries from, and liaising with, the contractor became critical. Senior management justified their decision in terms of the strategic importance of the new software. This decision was resented within ITD and the internal developers believed it made their job harder and less effective. Another important factor was the existence of a previous system; this meant that a major part of the work involved the conversion of old code to new, and ensuring that the old and new systems had satisfactory interfaces.

The project team and its organization was created by the Project Manager, helped by one of his early recruits, the group leader responsible for queries. They selected individuals from elsewhere in the company that they wanted to work on this project, mostly from within ITD, since most of the expertise they wished to capture for this project was technical. Some ex-users were also recruited from the Finance department. Recruitment was constrained both by the availability of people within ITD and by the project's ability to attract staff. As in Case 1 above, we believe the selection and recruitment of different types of expertise and knowledge reflects the key actors' views both of their task generally (developing software) and of the particular requirements of this project. Again we speculate that the views and representations held by these individuals are critical to the design of this project organization and reflect both their professional background and training as well as custom and practice within the company. Again it is interesting that certain sorts of expertise were not selected onto, or organized into, this development process. The most obvious omissions concern the re-engineering of the process of charging customers and its socio-technical redesign.

In Case 3 the new method of involving users was initially designed by a senior developer who was dissatisfied with traditional methods of developing systems, and was determined to find a way of involving users more directly in the process of software development. He obtained the support of his manager, the head of IT, and sought help from the Human Factors group who themselves had a professional commitment to user participation, and from the staff in Network Services who had good relationships with users in local offices.

This working group was able to obtain the support of senior managers responsible for the new software. This itself represents a considerable achievement in a large highly differentiated and political organization of this kind. However, it is fair to say that these senior managers were more concerned with getting the new software developed than they were with the new method of development. In

contrast to case study 1, the new method was developed 'bottom-up'. This helped account for the high levels of ownership and commitment to this new method of working from those at lower levels and most closely involved in it. The new method of user participation required the active support of three main groups, users, developers and the Network Services group. Their active support was essential for the new method to be implemented and stand any chance of success. It was also necessary to gain support from senior management for this 'bottom up' approach. Some felt that support from senior management could have been more active, and that this would have helped the implementation process.

In all three cases these project organizations were designed by their managers based upon their perceptions of required expertise and knowledge. People were selected into these development systems for what they could offer. The design of these systems, we believe, reflects their managers' views both of software development generally and of the demands facing their projects in particular. None of the actors in these systems advocated attention to social and organizational issues, i.e. a more socio-technical approach. Nor was there evidence, except in Case 3, of a focus on user-centred design or on education. We have argued that these are largely seen as technical enterprises and we speculate that this perspective arises from two sources. First, the people who design these systems are technologists themselves having experienced a professional training focused on technical concerns. And second, the companies themselves have particular ways of organizing and structuring these activities. There are some complex social processes at work here. For example, people trained in a particular way design a development system consistent with their background. They select into such systems similar people. These actors confirm and substantiate a particular set of perspectives over time. These are social systems, constrained by the perceptions of the key actors, the people on the team, their skills set, and the variety of expectations created for them by their super- and sub-ordinates.

7.4. *What disrupts the pattern of work organization? And with what effects?*

In Case 2 there were two main factors which disturbed or impacted upon this work organization. We have selected these two examples in part because they are different; one is concerned with something management did, the other with what management did not do.

In the first instance, all the project team members were given their appropriate CASE tools and trained in the one they would use and in the immediately preceding and following tools (as described in study 1). There were three principal effects of this implementation and the way in which it was managed; these reinforce the points made in case study 1 above. First, use of the CASE tools made the developers more specialized, as defined by the tool they used. This job specialization was determined not simply by the technology *per se*, but came about as a result of management's choice over how the new technology would be used. The underlying managerial logic was that the new tools require substantial new expertise, and that the tools themselves and training on them are very expensive—therefore they required that the users be specialists in a particular tool, rather than generalists trained in and using several or all of the tools. Second, use of the new tools led to the emergence of a new role, that of 'designer', to cover the intermediate stage between analysis and programming. Hitherto this role had not existed in this company; the analysts and programmers had worked together in an informal negotiated way to undertake design. And third, the ways in which programmers worked became disrupted by use of the new tools. For example, the group leader responsible for the conversion group described how a 'skeleton' program would be developed first and gradually filled in until that component of the program was completed. This allowed the programmer to develop an overall understanding of the interrelationships between different programs and modules before detailed design took place. The introduction of CASE disturbed this practice because it forced the programmers to write complete program modules before trying them out; the programmer was required to specify a great deal of detail into the tool that undertook automatic code generation. As such what previously had been a mixture of top-down and bottom-up planning, was replaced by an explicit commitment to top-down specification. This disturbed the normal practice of programming where a mixture of the two is commonplace (Davies 1991).

The second example is of a different order and reflects the failure by senior managers either in the user department (Finance) or in the project team, to consider business process re-engineering the process of charging customers. In practice the Finance Department argued for as much flexibility as possible in charging protocols using the logic that the system should accommodate whatever services and arrangements are negotiated with the customer. According to our informants, this meant that the software proved to be enormously complex, resulting in the development process taking 10 times longer than was originally estimated. This was reflected in the final cost of the system. No one challenged this view, nor argued for some simplification prior to new system development.

In Case 3 three principal factors 'disturbed' the original method of working. First, one of the senior managers within Network Services was concerned that the new development was taking a great deal of time. The original analysis work had indeed taken several years. Even more important however, he was aware that the development project was in

real danger of replicating the old system, providing a more automated version of what currently existed. He had recently completed an MBA thesis on the subject of business process re-engineering and he was determined that the current methods of working should be analysed and simplified, prior to any further computerization. He also felt that there was insufficient expertise within his own organization to undertake such an exercise in the necessary time. As such he hired a group of management consultants to assess the opportunities to business process re-engineer the work area in which the new system was to be applied. At the end of this work the consultants reported their conclusions; they reported that there were opportunities to re-organize some of the work, and also to increase the functionality of the new computer system. When the consultants reported their findings to senior management they argued that they could write this new software, and also deliver it sooner than could the in-house development team. Senior management gave them a contract and thus the consultants became a partner in the new development process. This second 'disturbance' put a number of pressures on the development process. A group of new developers from the external consultancy were introduced to work alongside the existing developers and users, but this new group had no commitment to user participation and no involvement in the Cooperatives or User Groups. This meant that whole areas of the new system were not open to the new form of user participation. To these consultants, users participated by helping provide a set of requirements and by acceptance testing. For their part, the in-house developers did not object to the principal of business process re-engineering. Indeed they felt they themselves could have accomplished what was required. Relationships between the in-house developers and the consultants were rather cool.

And finally, the involvement of the consultants led senior managers to bring forward the deadline for delivery of software. This meant that the timescales were even tighter and put the whole development process under a great deal of pressure. This was the major reason why the manager of development decided not to reinstate the Cooperatives and to remove direct contact between developers and users. The impact of the pressure to charge ahead with development was that the User Groups were following in the developers' slipstream, often undertaking evaluation work on old versions of the system. In practice the work of the users and the developers became uncoupled and asynchronous. This proved very difficult for the Network Service staff to manage and became wasteful of resources and effort.

In addition to these problems within the development area, the other major difficulty concerned the lack of organizational integration between the various aspects of development. The original analysis work was done by one group, the business process re-engineering by another, and the development work by a mix of internal and external developers with some support and input from users. Whilst the new software was delivered according to the specified schedule, the development process could not be described as effective.

One clear finding from these cases is that these work organizations are regularly disrupted and disturbed. Indeed, one of their apparent strengths lies in their flexibility and the capability they have for handling uncertainty and complexity. Interestingly for us, the major apparent disturbances in these cases all arose outside the development teams. In most cases, they resulted from the actions, and in some cases, inactions, of their senior managers. The introduction of CASE tools, the failure to undertake BPR, the decision to undertake BPR using external contractors (near the end of development!), outsourcing part of software development, bringing forward delivery deadlines—these were the major factors impinging upon the conduct of the work. And all were the result of decisions (or lack of them) by senior managers.

We conclude these are volatile, uncertain and fluctuating environments. Many of these changes represent severe disruptions to working. These managerial decisions have unanticipated and unplanned consequences for those working on software development projects. For example, in Case 2, the introduction of CASE tools affected the work organization and role specialization of the developers. They believe this was not good for them—and in the longer run, it will almost certainly not be good for the systems they are required to develop. In Case 3, the decision to use external consultants led to the partial collapse of the user-centred method of system development. One clear lesson here is that we should not be making software development processes more partial in the knowledge and expertise they include. This is, of course, not to argue that senior managers should leave software development systems alone—quite the reverse, since we believe some changes are required. But rather, we would argue that external decisions should be analysed for their internal impacts on the process and organization of software development.

8. Implications

We wish to make a number of interrelated points. First, the old modes of thinking about the organization of work were dominated by the archetypal blue-collar production line. Newly emergent forms of working are more likely to be project based, involving temporary teams incorporating different forms of expertise working flexibly in the face of high levels of uncertainty. These are knowledge-intensive work organizations. Knowledge and expertise are at the heart of the design of these systems. Software development

teams, for example, are designed specifically to include multiple forms of knowledge and expertise. Our case studies have demonstrated that an understanding of the forms of knowledge and expertise that were included and excluded, and the ways in which these inputs were or were not coordinated, helps explain some of the major difficulties with their processes and outcomes. We speculate that such forms of work organization will become increasingly common. We would add that the notion of knowledge work should also be extended to more traditional forms of work, including manufacturing.

Second, we believe that work organization theory is currently ill-equipped to address these organizational forms. Current research evidences a lack of attention to the derivation of different work organizations and to a detailed understanding of their actual workings. But more than a straightforward reorientation is required. Adopting a perspective which places knowledge and expertise as issues central to design and practice, makes theoretical and empirical demands of researchers. For example, it raises questions such as: What forms of knowledge and expertise are designed into (and out of) work organizations? Why? How are the different forms of knowledge and expertise organized? How does knowledge and expertise get shared and developed? These questions can only be addressed by an understanding of cognition in action in organizations.

Third, from an organizational perspective, Morgan (1986) was right to advocate the need to think in 'loops not lines', with a focus on the complex interdependencies within work systems. Another related idea occurs to us as a result of these cases. It may be fruitful to adopt 'loose' work organizations, ones that are 'minimally specified' in the means of achieving their goals. Our analyses suggest that work organizations comprising multiple forms of knowledge and facing high levels of uncertainty, may require internal loose organization, whilst being tightly coupled with their surrounding work systems (Peters and Waterman 1982). The new knowledge based forms of work are more likely to be 'loopy' and 'loose' in their organization, and our analyses need to reflect the requirements of a new mode of thinking.

Fourth, it is apparent from Case 2 that technologies do impact upon work organization. The introduction of CASE tools had a series of unanticipated effects on the organization of the process of software development. But this is not an argument for technological determinism. The introduction of new technology led to some fragmentation and specialization, but this was the result of how management chose to operate the technology, reflecting their views on the costs of the new CASE tools and the training required. Other choices were open to management involving other forms of work organization.

Fifth, a focus on the derivation and practice of work organization points to some very complex social and cognitive processes. We have argued that the design of work organizations is influenced by the cognitions of key actors, for example regarding software development generally, and the tasks involved in addressing a particular project. These perspectives are likely to be influenced by professional training, by custom and practice in the organization, and also by the complex web of views and actions of super- and sub-ordinates. In turn, the adopted work organizations help shape the roles of the actors within them, which thereby influence both their actions and their cognitions. At this stage we can only speculate on the potential for mutual shaping and interdependence in this area. More work is required teasing out some of these processes over time, though we recognize this represents a major challenge methodologically, empirically and theoretically.

Sixth, it follows from what we have said above, that there is a need for detailed and intensive case studies as an appropriate research strategy. These are complex natural histories that are only capable of description and analysis by certain styles of work, at least in the first instance. Such approaches are highly interpretive but likely to deliver more holistic understandings.

Seventh, we wish to draw attention to a number of practical speculations that arise from this work. We have argued at a general level that these are knowledge-intensive work organizations that require the coordination and integration of different forms of expertise and knowledge. Problems arise particularly when intrinsically important forms of expertise, most obviously of a non-technical nature, are excluded (as in Cases 1 and 2) or when they are not integrated in such a way that they constitute a coordinated effort (as in Case 3). There is not space here to describe our recommendations in each of these cases, but they each concerned the adoption of a more socio-technical perspective and the inclusion of explicit consideration of organizational structures and processes. As an illustration, in Case 3 we recommended the use of 'software development cells' using an idea from manufacturing industry. This would require the incorporation in a single workgroup of the full range of expertise and skills necessary to undertake a particular socio-technical design (including software development) from beginning to end. In all three cases senior managers accepted our socio-technical propositions; both organizations are keen to collaborate further and are developing plans to add a significant socio-technical emphasis to their future development projects.

Unfortunately we believe there is a real danger that the software development community more generally is moving in a quite different direction. There are three popular trends that give us cause for concern, namely: the increasing use of the factory production line metaphor for software development; the related belief that new advanced

technologies in the form of structured methods and CASE tools can standardize and integrate the work, even though they may fragment knowledge, expertise and understanding; and the fragmentation of the process of development by such approaches as outsourcing. We are highly sceptical of the wisdom of these initiatives because they make the integration of knowledge-intensive work harder. The reverse is needed, that is greater organizational integration of knowledge.

Eighth, we draw attention to the levels of performance achieved in each of our case studies. Whilst each of the systems was implemented and subsequently did function, none of them succeeded in fully meeting their objectives. The emphasis on technology, the failure to address wider organizational and human issues, the lack of inclusion of other forms of knowledge and expertise in the processes of development and implementation, jointly led to these failures.

Finally we should be clear what claims we are making of this work. We are not claiming that our findings are representative of software development practices generally. More work is needed testing out these ideas and conclusions. These cases were undertaken in relatively large and successful organizations, operating at the forefront of work in the field of IT. Many others are likely to be less effective. We also believe the integration of knowledge regarding technical and organizational issues could be even harder to achieve in the case of software development by specialist organizations for use in other companies. We should also be clear what claims we are making scientifically. We believe it is not possible to understand the practices, outcomes and derivations of work organization, in the field of software development or indeed logically in any other, without recourse to both organizational and cognitive explanations. Most obviously these projects and processes were designed the way they were to reflect the cognitions of the managers concerned. The work organizations were shaped by managerial and other cognitions. But at the same time, these cognitions were themselves shaped by organizational circumstances. It is the dynamic nature of these interplays that represents a major challenge for those of us trying to work in this field. One goal is a more holistic understanding of the derivations, practice and outcomes of work organization; another is improved integration between parts of organizational and cognitive psychology.

References

Axtell, C. M., Waterson, P. E. and Clegg, C. W. 1995, User participation in systems development: The importance of organisational factors, in S. Robertson (ed.), *Contemporary Ergonomics '95* (Taylor and Francis, London).

Blackler, F. and Brown, C. 1986, Alternative models to guide the design and introduction of new technologies into work organisations, *Journal of Occupational Psychology*, **59**, 287–313.

Broadbent, D. E. 1993, Planning and opportunism, *The Psychologist: Bulletin of the British Psychological Society*, **6**, 54–60.

Cherns, A. B. 1976, The principles of socio-technical design, *Human Relations*, **29**, 783–792.

Cherns, A. B. 1987, The principles of socio-technical design revisited, *Human Relations*, **40**, 153–162.

Clegg, C. W. 1994, Psychology and information technology: The study of cognitions in organisations, *British Journal of Psychology*, **85**, 449–477.

Clegg, C. W., Waterson, P. E. and Carey, N. 1994, Computer supported collaborative working: Lessons from elsewhere, *Journal of Information Technology*, **9**, 85–98.

Curtis, B. 1988, Five paradigms in the psychology of programming, in M. Helander (ed.), *Handbook of Human–Computer Interaction* (Elsevier Science Publishers, North-Holland).

Davies, S. P. 1991, Characterizing the program design activity: neither strictly top-down nor globally opportunistic, *Behaviour and Information Technology*, **10**, 173–190.

Eysenck, M. W. 1984, *A Handbook of Cognitive Psychology* (Erlbaum, Hove).

Eysenck, M. W. 1990, *The Blackwell Dictionary of Cognitive Psychology* (Blackwell, Oxford).

Eysenck, M. W. and Keane, M. T. 1990, *Cognitive Psychology: A Student's Handbook* (Erlbaum, Hove).

Flor, N. V. and Hutchins, E. 1992, Analyzing distributed cognition in software teams: a case study of collaborative programming during perfective software maintenance, in J. Koenemann-Belliveau,T. Moher and S. Robertson (eds), *Empirical Studies of Programmers: Fourth Workshop* (Ablex, Norwood, NJ), 36–64.

Foster, M. 1972, An introduction to the theory and practice of action research in work organisations *Human Relations*, **25**, 529–556.

Frese, M. and Zapf, D. 1994, Action as the core of work psychology: A German approach, in H. C. Triandis, M. D. Dunnette and L. M. Hough (eds), *Handbook of Industrial and Organizational Psychology*, Vol. 4 (2nd edition), (Consulting Psychologists Press, Palo Alto, CA).

Hacker, W. 1985, Activity: a fruitful concept in industrial psychology, in M. Frese and J. Sabini (eds), *Goal Directed Behaviour: The Concept of Action in Psychology* (Erlbaum, Hillsdale, New Jersey).

Hackman, J. R. and Oldman, G. R. 1976, Motivation through the design of work: Test of a theory, *Organizational Behaviour and Human Performance*, **16**, 250–279.

Hutchins, E. 1989, The technology of team navigation, in J. Galegher, B. Kraut and C. Egido (eds), *Intellectual Teamwork: Social and Technical Bases of Collaborative Work* (Lawrence Erlbaum Associates, Hillsdale, NJ).

Hutchins, E. 1991, Organizing work by adaptation, *Organization Science*, **2**, 14–39.

Karasek, R. A. 1979, Job demands, job decision latitude and mental strain: Implications for job redesign, *Administrative Science Quarterly*, **24**, 285–308.

Mintzberg, H. 1979, *The Structuring of Organizations* (Prentice-Hall, Englewood Cliffs, NJ).

Morgan, G. 1986, *Images of Organization* (Sage Publications, London).

Olson, G. M. and Olson, J. R. 1991, User-centred design of collaboration technology, *Journal of Organizational Computing*, **1**, 61–83.

PARKER, S. K. and WALL, T. D. 1995, Job design and modern manufacturing, In P. B. Warr (ed.), *Psychology at Work*, 4th edition (Penguin Books, London).

PETERS, T. J. and WATERMAN, R. H. 1982, *In Search of Excellence* (Harper and Row, New York).

van EIJNATTEN, F. M. 1993, *The Socio-Technical Systems Design (STSD) Paradigm* (Graduate School of Industrial Engineering and Management Science, Eindhoven).

WALL, T. D. and JACKSON, P. R. 1995, New manufacturing initiatives and shopfloor job design, in A. Howard (ed.), *The Changing Nature of Work* (Jossey Bass, San Francisco).

WATERSON, P. E., CLEGG, C. W. and AXTELL, C. M. 1996, submitted, The dynamics of work organization, knowledge and technology during software development, Institute of Work Psychology, Sheffield, IWP Memo No. 4.