ELSEVIER

Interacting with Computers 13 (2001) 447–466

Interacting with Computers

www.elsevier.com/locate/intcom

Time and representational devices in Rapid Application Development

D. Tudhope[a,*], P. Beynon-Davies[a], H. Mackay[b], R. Slack[c]

[a]*School of Computing, University of Glamorgan, Pontypridd, CF37 1DL, UK*
[b]*Faculty of Social Sciences, The Open University in Wales, 24 Cathedral Road, Cardiff, CF1 9SA, UK*
[c]*Research Centre for Social Sciences, University of Edinburgh, Old Surgeons' Hall, High School Yards, Edinburgh EH1 1LZ, UK*

Received 11 December 1999; revised 15 July 2000; accepted 12 September 2000

Abstract

This paper discusses an ethnographic study of a commercial prototyping software development project. A distinguishing feature of the development was its concentration in one span of time and in one room, with both users and developers participating. This gave rise to a working practice based around the use of low technology representations of design. The case study explores practical issues important for prototyping: time management, user involvement, everyday design representations and the development environment. The mundane nature of design representations facilitated user participation. The public representations of work on the walls showed the current state in design of different components of the system and facilitated collaborative activities. The case study was part of a larger research project (1995–1998) which investigated the commercial use of prototyping in the UK. The development was influenced by a recent trend in commercial prototyping practice, Rapid Application Development (RAD). Implications of the case study for RAD and participatory design are discussed. © 2001 Elsevier Science B.V. All rights reserved.

Keywords: Prototyping; Ethnography; Rapid Application Development; User participation; Design representations

1. Introduction

An emphasis on the user and prototyping has been one of the key contributions of HCI to systems design (e.g. Gould and Lewis, 1985; Hartson and Smith, 1991). Putting this emphasis into practice, however, can pose pragmatic and organisational problems. Issues

* Corresponding author.

E-mail addresses: dstudhope@glamorgan.ac.uk (D. Tudhope), pbeynon@glamorgan.ac.uk (P. Beynon-Davies), a.h.mackay@open.ac.uk (H. Mackay), rslack@afb1.ssc.ed.ac.uk (R. Slack).

PII: S0953-5438(00)00050-3

relating to user involvement, design representations and time management come to the fore in prototyping projects. The application of ethnographic techniques from anthropology and sociology to the study of user work practice in order to inform design has been a recent trend in HCI (Anderson, 1994; Blomberg et al., 1993; Suchman, 1987).[1] Hughes et al. (1993) provide an overview and discuss the potential of the resulting rich descriptions for uncovering implicit practices which might be invisible or taken for granted in formal descriptions of a work setting. Some of this work has discussed representational devices in the context of command and coordination activities, such as airport control operations (Heath and Luff, 1992; Hughes et al., 1992; Suchman, 1992). An ethnographic perspective is also applicable to the study of how computing design representations are used in practice (Karat and Bennett, 1990, 1991; Muller, 1993). The particular features of our case study, which had a high degree of user involvement, gave rise to a development practice based around the use of low technology representations of design. Time has recently emerged as an issue for HCI (Fabre and Howard, 1998). In this paper, our focus is on time as it figures in the iterative development process, as opposed to interface issues. ‘Shorter time to deliver’ was identified as a ‘prime focus’ by managers in our case study company. Paradoxically, meeting this need required that a fairly significant part of project time be spent on time management and co-ordination activities.

The case study takes an ethnographic approach to a commercial prototyping development and discusses its wider implications. This study is part of a larger multi-disciplinary research project, involving computer scientists and sociologists, which investigated the commercial use of prototyping in the UK during the period 1995–1998. The first stage of the project undertook a literature survey and a series of semi-structured interviews with developers on the role of prototyping in systems development (Beynon-Davies et al., 1999a).[2] Although prototyping has been used in various forms for many years, a feature of our initial survey was the current interest and popularity in the UK of RAD (Rapid Application Development) in commercial organisations. We found that inquiring about prototyping often elicited a response in terms of RAD. The case study, an internal project in an international utility corporation, was seen as a prototypical RAD project by the company.

The practical management of prototyping projects emerged as a key concern for developers from the interview phase of our research — issues including user and project management, proliferation of user requirements, excessive user expectations, and involvement of users (Beynon-Davies et al., 1999a; Tudhope et al., 2000). For example, one developer identified a problem with hacking style approaches to prototyping where projects could just ‘go on and on forever’. We found widespread employment of incremental, rather than throwaway prototypes, due to the commercial time pressures faced by our interviewees. For the developers in our wider project, the relevance of RAD and related techniques lies in a perception that today’s economic climate requires businesses

[1] Ethnographic studies focusing particularly on design/development work include Button and Sharrock (1993), Suchman and Trigg (1993) and Woolgar (1991); and within the Participatory Design tradition of action research, see Greenbaum and Kyng (1991).

[2] For other empirical surveys of prototyping, see Alavi (1984), Carey and Mason (1983) and Gordon and Bieman (1995).

Table 1
DSDM principles

1	Active user involvement is imperative
2	DSDM teams must be empowered to make decisions
3	Focus is on frequent delivery of products
4	Fitness for business purpose is the essential criterion for acceptance of deliverables
5	Development is iterative and incremental
6	All changes during development are reversible
7	Requirements are baselined at a high level
8	Testing is integrated throughout lifecycle
9	Collaborative and co-operative approach between stakeholders is essential

to deliver systems within a faster timescale than allowed by traditional methodologies. A frozen set of initial requirements is seen as unrealistic in many situations, since users often wish to make changes at a later stage in development, having had experience of the technical possibilities through exposure to prototypes (Bates, 1995; Harker, 1991; Mogensen, 1992). Notwithstanding, we encountered general appreciation by developers of the potential benefits of involving users when prototyping, in terms of greater fit with user needs, early feedback on misunderstandings and user ownership. The promotion of active user involvement in design work has been emphasised by various HCI researchers, via training and techniques to facilitate user input (e.g. Bødker, 1991; Ehn and Kyng, 1991; Karat and Dayton, 1995). Thus, the management of time and the user-centred representation of design activity are topical concerns for both developers and researchers. This paper explores these issues within the context of our case study and discusses broader implications for RAD and participatory design approaches.

Section 2 of this paper provides an overview of RAD while Section 3 introduces the case study. Section 4 then focuses on the key issues emerging from the ethnography: the low technology design representations employed in the project and issues of time management. Section 5 draws conclusions and discusses the implications of the research.

2. RAD overview

RAD coalesced as a methodology out of the experience of large US corporations in the late 1980s and was popularised in the commercial world by Martin's book of the same name (Martin, 1991). It has evolved in industry with little influence from the academic world. In contrast to highly structured waterfall methodologies, RAD subsumes a collection of principles and approaches. It might be characterised as an approach to project management. The DSDM (Dynamic Systems Development Method) Consortium was formed in 1994 by vendor and user organisations to produce and promote a public domain, non-proprietary RAD manual (DSDM Consortium, 1995). The nine underlying DSDM principles are detailed in Table 1.

DSDM emphasises the need for prototyping tools but is not prescriptive — 4th generation systems, object oriented toolkits, Relational Databases, rapid prototyping tools, Visual Basic and Perl have all been used. DSDM recommends a 'filter' be applied to determine whether a given project is appropriate for a RAD approach. Target systems are database-oriented applications that are not computationally complex, and where the interface is important. Stapleton (1997) gives a recent overview of RAD and DSDM while a broad review of RAD literature from an Information Systems perspective can be found in Beynon-Davies et al. (1999b).

Time is a prominent feature of RAD. A much-quoted RAD axiom is that 80% of a system (taken to include the system's key features) can be produced in 20% of the time required to build the complete system. This has led to the concept of 'timeboxing', and a change in the relative priority of time and requirements compared to traditional methodologies. If a deadline is in danger of being missed, lower priority requirements are moved back to a later timebox, or a possible later iteration through the lifecycle. This requires an initial prioritisation exercise on user requirements. The emphasis is on delivering a version of the system within a short, fixed timespan before the business environment has changed significantly.

> Without timeboxing, prototyping teams can lose their focus and run out of control. … Timeboxing means setting a deadline by which a business objective must be met, rather than describing when a task must be completed. To be effective, a timebox (which might deliver, say, a complete part of the system or an early version of the system) should last a maximum of six weeks and typically would last no more than two weeks. … At the deadline the user must be able to approve the delivery of the prototype covered by the timebox. If it appears that deadlines will be missed, the deliverable should be de-scoped so that it can be produced successfully in the time available, i.e. requirements can slip, timing never does. (DSDM Consortium, 1995, p. 70)

According to the DSDM manual, there are two variants of the project lifecycle; one is an 'ultra-intensive' development with all RAD team members present, in one place for a fixed period, as in the (three week) case study discussed here. A more common alternative is a three to six month development, with associated two-day JRP (Joint Requirements Planning) workshops, and more focused, shorter JAD (Joint Application Design) sessions. In DSDM, a distinction is made between early requirements gathering JRP workshops and later prototype evaluation JAD workshops. It is important that customer (user) representatives attend each type of workshop. Depending on the project, there is sometimes more emphasis on management representatives attending JRP workshops and end-users attending JAD workshops. It should be noted that in some related literature, both types of workshop come under a JAD umbrella (Carmel et al., 1993; Wood and Silver, 1989). Although influenced by JAD, RAD encompasses a wider view of the project lifecycle. As an example of the less intensive project variant, the lifecycle of one prominent RAD vendor organisation (not the focus of this study) consists of iterations around three main prototypes, the Navigable Model, the Pre-Production Prototype and the Production Prototype — fully functioning prototypes based on an object oriented database and developer environment. Our case study

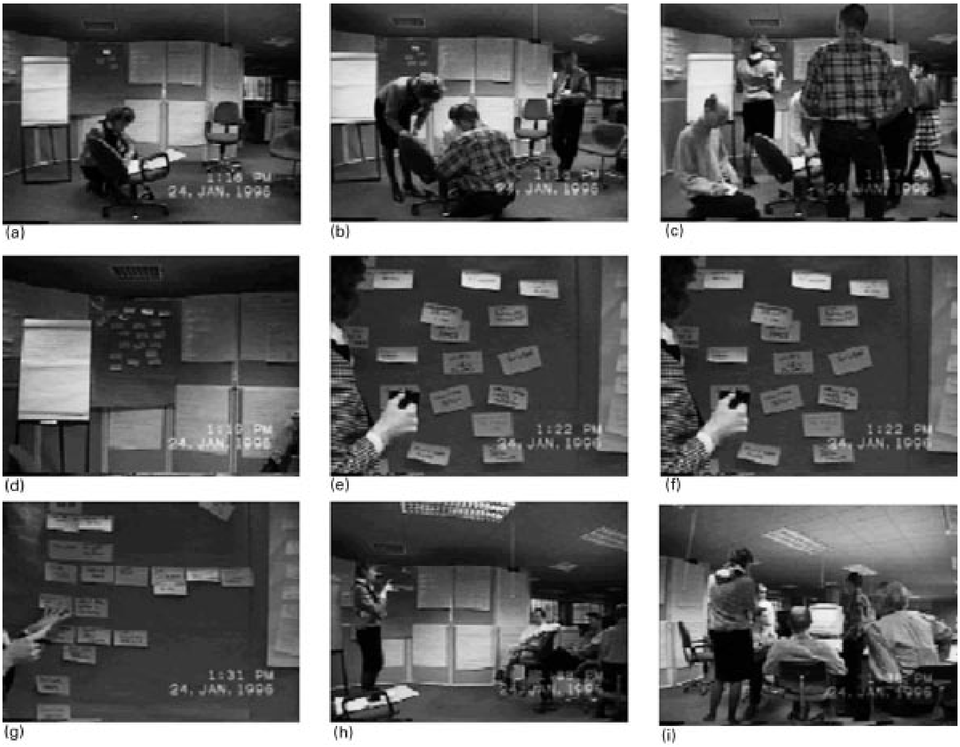

Fig. 1. ‘Brown papering’ a mock-up.

represents the less common variant of the lifecycle, but within the case study company culture it was considered to be a very ‘pure’ RAD example, due to the intensive user involvement and the 100% focus on a single project.

In sum, distinguishing features of RAD are prioritised requirements, timeboxing, evolutionary prototyping with active user involvement and rapid development tools. One of our respondents succinctly defined RAD as ‘empowered, cross-functional teamworking to strict deadlines in a clean room environment’. The ‘clean room’ is a feature of the intensive form of RAD development, being an area set aside for the purposes of the development with little chance of interruption. The DSDM manual emphasises the need to educate customer organisations on the importance of user involvement and the need for joint approaches to project management. Change requests should not just ‘drop out of the sky’, as a developer put it at one of the case study company’s lunchtime talks. Higher management support is considered important and different user roles are suggested, including management roles, end-user roles and the Ambassador User who acts as a spokesperson for different user groups and is empowered to take decisions.

3. Case study of an intensive RAD project

The case study was an in-house intranet development carried out by the corporate relations department of a large international utility corporation, over a three week period in 1996. The corporate relations department deals with all press and public relations activity and has various centres in the UK. The aim was to create an internal website that would act as a resource base for coordinating public relations activities and campaigns and avoid duplication of effort by different centres across the country. This would allow online access to corporate information, including a diary system for corporate relations staff, together with a project management 'manual' allowing evaluation of proposed public relations projects against corporate objectives. A JRP workshop, including higher management, had generated a prioritised list of requirements and a risk assessment, before our involvement with the project. Essentially, the development consisted of creating an extensive set of web pages with associated Perl scripts, building on but significantly extending an earlier RAD intranet development.

The corporate relations department chose an intensive development approach, in other words the project was concentrated in a short span of time, at one site, with user representatives always at hand and ostensibly removed from outside distractions (in a clean room). In fact, the development and its predecessor were considered flagship projects within the organisation by champions who were advocating increased use of RAD. The setting was a large open-planned office in a city-centre, with a section closed off for the project by a number of room dividers (see Fig. 1a). The project leader (a user) was a senior member of corporate relations with experience of the previous development. The two developers, who both had some experience of RAD, were from the organisation's computing division and lived in the city (but did not normally work in the setting). The four user representatives, sometimes referred to as 'the customers', were drawn from various UK centres. They stayed at a hotel during the week and commuted home at weekends. Two users were PR officers in their usual work and two were 'librarians' (of PR material). The two developers made use of Perl on workstations, while each user had access to a PC with office applications and HTML editors. Daily 'wash-up' meetings (see Section 4.3) reviewed progress and allocated tasks. Work tended to tail off on Friday afternoons, while users usually spent Monday mornings dealing with issues at their normal work places via telephone. In the first week, the project manager was concerned to explain the process and encourage user participation and input. The second week saw more emphasis on the completion of development tasks. Visits by management and ensuing demonstrations augmented this general pattern. For example, an intervention half-way through the project by management representatives, not members of the design team, resulted in some changes to the proposed design. This was scheduled as a formal review in the project plan and was motivated by strategic concerns for the project. It might seem to go against the RAD philosophy that design teams be 'empowered' to make decisions independently. However, such guidelines are necessarily applied within organisational constraints. From our wider research project, developers influenced by RAD ideas tend to apply the guidelines in a contingent manner, according to context, and it would be rare to meet a project that fulfilled all DSDM prescriptions. In this case, the intervention resulted in some specific design changes with consequences for immediate tasks but did not appear to affect significantly the design process or time schedule.

The ethnographer observed the development, videotaped, conducted informal interviews, and participated in the lunchtime and evening social activities. An index of the video material was compiled subsequently. From this, in conjunction with field notes, selections of video material were identified for closer study. The project was notable for the active involvement of users in design and development activity, stemming from the decision to commit user representatives to the three week period. (This also raised some difficulties which are taken up below). The consequence was that feedback on prototypes could occur frequently and informally. Demonstrations were scheduled as part of the day's activities but they also happened informally, sometimes by an invitation broadcast across the room. Demonstrations frequently followed a co-operative prototyping approach (Bødker and Grønbæk, 1991) with live changes made by developers using Perl libraries in response to user feedback. These demonstrations were also able to deal with issues of work practice, since users were present. For example, one demonstration led into a discussion on issues of passwords, personal homepages and provision of a facility for users to see whether 'their name is being taken in vain elsewhere' — there was a concern that tasks might be surreptitiously assigned to subordinates with the new project planning functionality.

It would be wrong to assume that the users spent their time waiting to give feedback. Some were involved with editing HTML documents, very much part of the development activity in this particular project. A considerable amount of time was also spent in mock-up design activity and in daily planning meetings. The team ate lunch together and informal discussions on project related matters were typical, although the developers occasionally took lunch separately in the city. Since members were located in an area away from their home sites, they socialised with each other most evenings, drinking and eating together, sometimes subsidised under the rubric of 'team building'. The ethnographer was often included in these events, and noted that the group formed its own points of common ground such that on some occasions there were clear notions of an 'us' and a 'them'. The boundary was not between users/developers, but in-team/outsiders. Engendering team spirit is seen as a necessary step in replacing the 'traditional safety net of… the signed-off specification' (DSDM Consortium, 1995) by joint responsibility for any problems that develop.

4. Low-technology design representations and time management

4.1. Public project memory

The intensive nature of the project in one development room with developers and users continually present, had consequences for the organisation of work. The material environment could be organised around the needs of this one project. Temporary posting of flip-chart pages is relatively common in group workshops. However, since the team had exclusive use of the development area for three weeks, they were able to leave action item ‘to do’ sheets posted on the ‘walls’ (room dividers) from day to day (as seen in Fig. 1a). These A2 sheets with tasks to be done and the allotted team members were public representations of project activities. The initial list of requirements was also posted (and subsequently annotated). The project leader’s introduction on the first day, part of an overview of RAD and allocation of responsibilities for the project, gives the rationale for the use of the walls as collective project memory:

> We don’t want to have to write things down, waste of time — pages and pages of procedures and things. It goes on the wall. We’ll just throw it on the wall and that’s documentation (apart from the developers who’ll have to do a bit more than that). So there’s a page each day. At the end of each day, we’ll go through the list for the day and carry forward items that we’ve started. The deadlines will go up here. If there’s something happening, just throw it on the wall so it’s not forgotten. Anything at all just put it on the wall, so we’ll know where it is.

Part of the context to this introduction is the RAD philosophy that, to some extent, a prototype development environment is self-documenting. Interaction between users and developers is encouraged, reducing the amount of time spent in formal documentation activity in order to facilitate rapid development and feedback on prototypes. The issue of documentation is debated in RAD literature and a concluding documentation phase is sometimes recommended for RAD projects.

4.2. Mock-ups and brown-papering

> It’s lots of bits of paper and ticks and post-it notes and things… you’ll get to understand it; it’s not *that* bad really.

Here, one of the users in the case study explains to a newcomer (one user was replaced half way by another user representative due to other duties) how things are done, as part of familiarisation with the project and current design. The quotation illustrates how a non-technical user’s apprehensions and preconceptions about design activity are hopefully assuaged by the mundane nature of the technologies involved. Users actively participated in important aspects of the design work. In part, this was doubtless due to the less technical web design component of the development in this project. However, it was also due to the use of everyday design materials, requiring little or no training. A notable aspect of the project was the use of annotated post-its[3] as (easily movable) mock-up design elements on flip-charts or even brown paper sheets. They were used both for screen design and to organise the hyperlink structure and architecture of the system.

Fig. 1 represents 22 min of a design session early in the project, where components of the system functionality are represented by post-its. At the start of the session, the initial information resides in the project leader’s (private) notebook. Fig. 1a–c show team members (users and developers) collectively transcribing from the notebook to post-its and placing them on a brown paper sheet on the room divider. This public representation

[3] Post-It is a trademark of 3M Corporation.

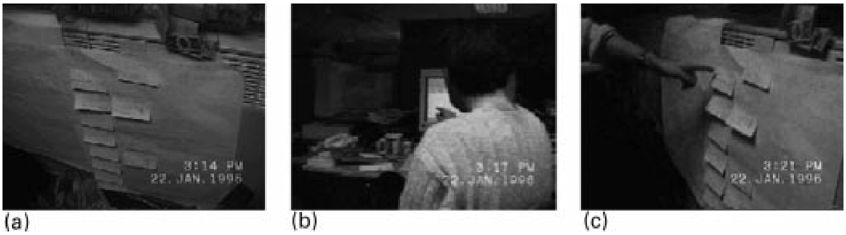

Fig. 2. Mock-up alongside developer workstation.

now affords collaborative activity. The initial result (Fig. 1d) was a chaotic arrangement, which was gradually transformed into an ordered representation of modules during a design session led by the project leader (Fig. 1e–g), with participation from team members sitting around the area. Post-its were re-organised and new ones generated during the discussion. In the final part of the same session, design activity turned from the paper mock-up to a discussion based around a prototype on a terminal located in the same area (Fig. 1h and i). The clean room environment allowed activity to flow easily from paper-based public representations to discussions around software prototypes.

Mock-ups were produced as part of team discussions and also as outcomes of small group work. In turn, they informed the wash-up sessions and acted as inputs to development work by the programmers. They were quite concrete design artefacts, sometimes being moved from one location to another, and acted both as outputs from and inputs to design work. One discussion on the diary design by two of the users and the project manager, generated a flip-chart with post-its. The project manager concluded by noting that the programmers 'can hang that on the wall behind them and work to that'. Fig. 2 shows just such an example of a programmer's work area with the mock-up attached to the radiator beside the computer. Here a discussion with several team members present is taking place around the workstation and at different times reference is made by the developer to onscreen elements and also to the paper mock-up. Fig. 3 gives an example where a mock-up, attached to the walls as a public representation is removed to a workstation to inform two users' development work. Conversely, in another session a small group design discussion started around an interactive prototype and then moved on to the walls and generated a new flip-chart. The public display of paper-based design artefacts encouraged collaborative activity with the users.

The term 'brown paper' has some specific significance. From our interviews and the RAD workshops we attended, 'brown-papering' is used to describe brain-storming or design activity, where rolls of brown paper flexibly yield more area for writing than is possible with a flip-chart's discrete pages. The ethnographer subsequently attended an in-house RAD awareness workshop, organised by RAD facilitators in the same organisation. At this meeting, as part of a response to a question on planning tools for rapid development, a speaker remarked:

> Perhaps one of the best planning tools that I know is a roll of wallpaper. You can stretch it out and put post-its onto it and colour it in. The point it gets across is that it

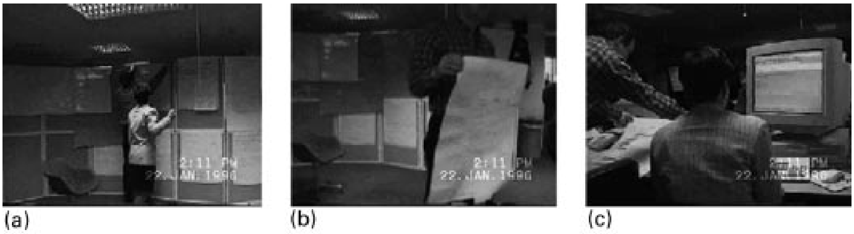

Fig. 3. Moving a mock-up to a workstation.

> is everybody’s responsibility for project management… it should be that you finish a part of the project and then you go to the paper and see what else you can pick up, that is how it should be done. I know that ⟨colleague of speaker⟩ saw some of this on ⟨the ethnographer’s⟩ video tapes, and that is really good.

Note the point that it is everyone’s responsibility for project management.

4.3. The management of time

The tasks to be performed each day were generated at a team meeting, known as the wash-up session, which concluded the previous day’s work. Here progress was discussed and members of the team reported on the tasks they had done that day, what more they had to do, and any problems. As well as a forum for the discussion of progress, the wash-up sessions also allowed the team to write a daily action plan. Each day, a flip-chart ‘to do’ sheet was completed by the team leader noting what had to be done the next day. In reporting on their day’s work, team members oriented to the retrospective/prospective character of the wash-up meeting. Members responsible for a section of the project were asked if their task had been completed. New action items were allocated to a team member, usually by team members volunteering for tasks. Names (some components required collaboration) were added to the action item. Each sheet was placed on a room divider, so that everyone could see what had been done and what was outstanding within the course of a working week.

The ‘to do’ sheets were publicly available within meetings, but they also stood as the record of a meeting and as a resource for team members to examine. Routinely, team members examined the sheets on entering the work area each morning to ascertain their tasks for the day. Where a task required collaboration, this would be discussed in the informal talk that occurred each morning. This frequently involved explicit reference (gestures) to the flip-charts. (The ethnographer’s attention was drawn to these organising artefacts when a team member remarked one morning: “Look at how we come in in the morning, and there’s always something on the bloody boards for us to do.”)

In wash-up meetings, the to do sheets and public design representations acted as the ground for collaboratively generating the next day’s tasks. As a new sheet was filled out, the previous sheet, detailing the current day’s work, was ticked off. Fig. 4 illustrates 38 min of the wash-up session on the day following the brown-papering session discussed in Section 4.2. The project leader ticks off action items completed on the current day’s

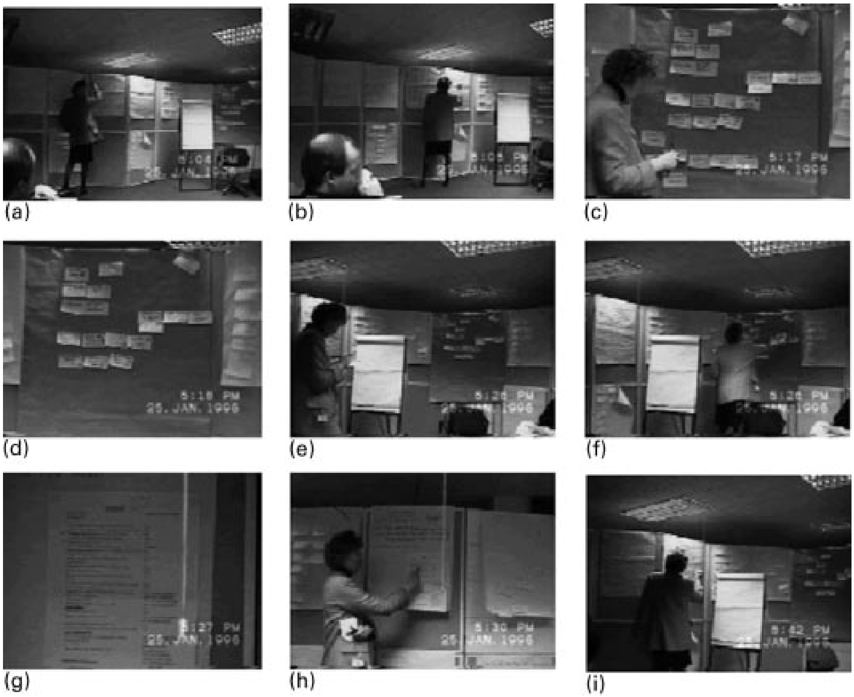

Fig. 4. The wash-up session.

sheet (Fig. 4a), and enters tasks with names of team members on next day's sheet (Fig. 4b). Then the design mock-up constructed the previous day (Fig. 1) is used to structure discussion. As an aid to developing the next day's activities, the mock-up is now partially disassembled (Fig. 4c and d). A new item is then added to the mock-up in this session (Fig. 4e and f). The wash-up session concludes by considering how the initial list of requirements, also on the walls affects the next day's tasks (Fig. 4g–i).

The initial requirements list had been prioritised into three levels at the JRP meeting preceding the intensive phase of the development. On the Friday of the first week, a timeboxing exercise colour coded the initial list of requirements, in terms of those that could be completed in the remaining two weeks and those that should be 'descoped' from this iteration of the project. In fact, all Priority 1 requirements were met by the project apart from those that had been identified at the initial meeting as not being achievable in three weeks or as being dependent on other company activities. The prioritisation of requirements meant that the way had been prepared for the team to omit lower priority tasks to meet the deadline. Prioritisation of requirements is critical and the DSDM manual lays great stress on it, observing that users are likely initially to assign everything top priority. According to the manual, this must be resisted at all costs and levels of priority should be identified to provide some flexibility. The exercise serves to prepare users for future descoping decisions, reminds them that there are relative priorities within a development and helps to create a climate for a team-based approach. Timeboxing should be considered in combination with prioritisation. We return to this issue in the discussion on incremental delivery in the next section.

5. Discussion and lessons learned

In this section we discuss the efficacy of the techniques we observed and RAD approaches more generally by reviewing the general outcomes of the case study and placing them in the broader context of the company and more widely. Finally we draw conclusions from the key issues arising out of the case study.

5.1. Efficacy issues

The project we observed was a second iteration and major extension of an intranet development (involving a different team) the year prior to our involvement. A subsequent (third) iteration occurred about a year later. Our close involvement with the project ceased with the completion of the second phase of development. This incremental development process is typical of RAD approaches, as discussed in Section 2.

The development we observed finished on time, met all major objectives (see discussion on timeboxing earlier) and was installed after a training period. The project manager demonstrated a near final version of the system in the third week of the development at a national company forum in London and reported back to the team that the demonstration had gone well with only minor changes required. The system was subsequently used operationally. A report by a RAD facilitator on the third iteration characterised the second iteration (our case study) as a successful development in terms of business benefits, senior management support, customer satisfaction and application of technology. Features of the case study were mirrored in the subsequent iteration, which implemented an extranet extension to external public relations agencies world-wide. This suggests that the techniques were viewed as useful within the company. Daily wash-up meetings with users and developers were employed in the third iteration to track progress. The report, however, recommended that attempts be made to minimise the time involved and that there be two stages to the wash-up, the first users-only and the second stage involving both users and developers. The third iteration also employed the same clean room facility, although it seems to have been more difficult to bring people together for the entire period (due to other demands on time). Changes in the team composition and absence at certain times of people with specialist knowledge were regretted and it was emphasised that neither users nor developers are interchangeable. The team working was reported in positive terms, although a problem with too frequent interruptions of developers by users was noted. Some physical means of indicating the current interruptability of developers was recommended.

Moving further away from the case study but within the same organisation, another report on a different development that attempted to follow a RAD (DSDM) approach offers some useful elements of comparison. Initial resource estimates were that the development would take five people twelve weeks. In the event, it was completed in seven weeks by four to five people, while the customers signed off and were pleased with the outcome. The development team worked almost exclusively on the project.

However, user representatives were not permanently in attendance but visited the team at least once a week, a hot desk being set up for user visitors. A clean room area was corralled off within the large open plan home office and developers' normal phone numbers were diverted for the duration of the project (e-mail was answered outside core hours). Daily wash-up meetings (5–30 min) with the whole development team present took place in conjunction with weekly review meetings involving customers and Monday morning planning meetings. A team ethos developed and the team all agreed to work late into the evening on several nights to achieve goals. (The report light-heartedly lists the supply of fruit under team building.) 'Invasions' of the clean room by non-team members were a minor irritant to be managed sensitively. The report's conclusions emphasised the importance of accurately ascertaining team members' availability and taking account of leave and conferences. 'Small wins' should be highlighted so that progress could be recognised. Customers should be kept involved and their expectations managed. The clean room was thought to have worked well, in part because of minimum interruptions (telephone, e-mail, visitors). On the less positive side, the report mentions the long working hours and the need for time to be allocated for testing. Thus this project appears to have been partly intensive (on the developers side only) with an informal clean room (but with stricter control of telephone interruptions) and to have followed the same pattern of wash-up meetings as our case study.

Although summative judgements on developments and perceived benefits may vary with stakeholders (and their organisational interests), the system developed in the case study was used by the organisation and the prototyping techniques were seen as useful. More broadly, in the first phase of our research more than a third of our interviewees claimed to employ some RAD features in their prototyping practice. In one interview with a developer in a large corporation (not the case study company), we were told that the development group had spent some time searching for methodologies to use with developments based around prototyping, arriving independently at a fairly similar position to the DSDM approach — strong emphasis on user involvement but with controls to prevent going 'into continual development'. These included having some designated user roles from the start, a timescale for completion (and drawing the line on unnecessary functionality) and formal review points (but not exceeding three iterations):

> Although prototyping itself if you like is an informal method of developing, you need to have formal structure to it. … What we would at least insist on having is a nominated user representative or preferably a user group and then at pre-determined stages, we would then sit down and present the prototype to that group… And I know we struggled for quite a while to actually find a methodology that supported rapid development, because it typically flies in the face of all the formal methods that you've known in other formal development approaches. I think that what you have to do is get the balance right — you need to have the formal structure round it and you need to have formal review points in it and then you use the power of the tools to get you to those review points in as quick a way as possible.

It is difficult to draw specific design recommendations from the unstructured qualitative data of ethnographic fieldwork (Hughes et al., 1993). Although some of the practices we observed appeared to work well, our aim in this paper is to explore issues arising from the case study rather than advocate any particular method. For a group of people to accomplish a collaborative project, members need a clear, collective understanding of what has been done, what they are doing, and what they are going to do next, along with some measure of understanding what other people are doing. While this may seem to be stating the obvious, it highlights intersubjectivity as an achievement of the (usually taken for granted) work practices employed on any project. The apparent orderliness of system development is not prescribed in advance by the methodology, but is an outcome of the methods members use for collaborative working, achieving mutual awareness and intelligibility.

In the remainder of the paper, we reflect on the inter-related themes emerging from this case study: time management, user involvement, everyday design representations and the design environment.

5.2. Management of time

> Let us no longer waste the time I say.
> My lords, time wastes itself by night and day,
> …
> Seneca and philosophers of old
> Bewail time's loss more than loss of gold:
> "lost money is not lost beyond recall,
> But loss of time brings on the loss of all."[4]

In our project, temporal patterning exhibited a particularly fine-grained character, with the daily wash-up session continuously reviewing and adjusting the public schedule of activities. The wash-up meetings constituted a significant part of the total time spent on the project (sometimes lasting more than an hour each day). The tension between the contribution of wash-up meetings to project management and the time devoted to them is discussed in Section 5.1. The wash-up sessions provided a forum for the team to assess their trajectory and to establish as a group where they were in relation to the objectives that they had established, both on a day-to-day basis and within the remit of the project as a whole. Wash-up sessions served both a coordinating and a motivational function. The 'to do' sheet sometimes contained references to evening social activities, while on another occasion the last item read: 'Working late, all'. On another level, each evening the wash-up session retold the evolving story of the project, a narrative joining past and future trajectories. In an early wash-up session, a user noted that he had not felt that much had been accomplished, but on reviewing the walls and participating in the discussion, he observed, "I feel better now, but before dinner [midday] I didn't think that we'd got anything done."

As discussed above, the development was part of a sequence of iterations. Martin (1991) argues that business needs change substantially during a lengthy development and that meeting current business needs when the system comes into operation is a better aim than

[4] Introduction to the Man of Law's tale, The Canterbury Tales, G. Chaucer. Translated by Nevill Coghill, Penguin Classics, p. 139.

meeting a long-frozen specification. The advantages of a series of timeboxed iterations over one large project were also stressed in an interview with a RAD consultant (not from the case study company):

> For instance, one of the large projects is over four years. At the beginning they had a very detailed set of requirements and we persuaded them that they didn't really know what things might be like in five years time. What we could do for them was things like timeboxed contracts so they could get things delivered every nine months and keep up with what was happening. So that's one way, parcelling things up so that they can actually envisage the horizon.

Work in social psychology can inform our understanding of time management practices. McGrath (1990) maps out the temporal dimensions of collaborative group work. Key generic problems are temporal ambiguity (scheduling), temporal conflicts (synchronisation) and scarcity of temporal resources (time allocation). He discusses a series of laboratory experiments which found that the imposition of tight deadlines resulted in lower quality work and less time spent on interpersonal communication and group discussion of ideas. However, this depends on the particular time constraints and the nature of the tasks involved. Further work is required in differing environments where situations are more complex than controlled experiments. Real world applications may have factors that interact with temporal aspects. For example, Egger and Wagner's (1993) study of hospital surgery scheduling provides an insight into a very different set of constraints and temporal practices from (say) system development projects. In our case study, time pressure was combined with possibilities of descoping requirements that had been previously prioritised.

McGrath emphasises the temporal patterning of group working, drawing on the concept of (social) entrainment, the subjection of one process to another temporal pattern, normally used with regard to biological processes (circadian rhythms). For example, the effects of shift working can be discussed in terms of a disentrainment between the rhythms of work and social life. The time constraints introduced by a RAD approach should be considered in the context of an iterative philosophy of staged incremental delivery.[5] Seen in this light, the timeboxing prototyping technique is an entrainment device and an emphasis on timeboxing and shorter development times can be viewed as a move towards a closer social entrainment of the rhythms of the development lifecycle to the rhythms of organisational and business cycles. 'Loss of time' can bring on the loss of the rationale, environment and working practice for a proposed system as the business context changes and user requirements evolve.

5.3. Design representations and user involvement

Memory is not a passive scanning of a tape, but an active process frequently involving

[5] See also Graham (1989) in a non-RAD context and for a Participatory Design rationale, see Lilienthal and Zullighoven (1996).

an interaction with some representation of the past[6] and linked to the planning of future activities. We are all familiar with leaving notes to ourselves on doors or refrigerators. At another level, interface elements call forth the associated sequence of operator actions we might find difficult to remember out of context. Anderson (1994) discusses typical work desks, with their various piles of paper and notes as public 'mnemonic devices', triggering recall of the current state of parallel activities. Hughes et al. (1992, 1993) discuss the importance of publicly posted paper strips with flight details for air traffic controllers; the strips act as a memory resource to make sense of the current situation and also offer a trajectory for future action. The public representations of work on the walls in our case study showed the current state in design of different components of the system and oriented team members to their current tasks, what their next task would be and with whom. Temporarily posting material on walls is not uncommon in JAD (Carmel et al. (1993) and some design work (e.g. Wall and Mosher, 1994). However, as discussed in Section 4, the clean room environment afforded the means for collaborative design and project management activities. This aspect of the study has some similarities with the use of the walls in the '4-walls' user-centred design environment (Karat and Bennett, 1990, 1991), where different walls hold different aspects of the ongoing design work, such as requirements and design constraints. There is some commonality in use of the walls and the resulting implications, such as the need for a dedicated (clean room) project area. However, there are also differences. For example, in our case study user team members were involved in collaborative activities using the low-technology mock-ups, some of which were moved from public wall display to local postings beside a workstation (and vice-versa) due to the combined design and development environment.

The mundane, everyday nature of the design representations facilitated user participation. The 'It's lots of bits of paper and ticks and post-it notes and things' discussion (Section 4.2) illustrates how end-user representatives were drawn into the design process. Our study supports the potential of low technology design representations for user participation in design (Bødker, 1991; Ehn and Kyng, 1991). The comparison with similar but higher level 'equal opportunity' participatory design toolkits, such as PICTIVE (Muller, 1991, 1993), is instructive. PICTIVE, which has been employed in several projects for facilitating participatory design activity with users, has a toolkit which includes post-its but also more specialised plastic icons for interface elements and events. These allow more fine-grained specification of interfaces at the cost of some increase in cognitive overhead for the user. The question of the appropriate level of complexity of design mock-ups for different user groups deserves further study.

User involvement was seen as advantageous by both users and developers in the project. The clean room environment with its continuous access to users was also helpful for informally arranging feedback sessions on prototypes. On one occasion, a developer commented that a discussion with users during a demonstration led to a faster design solution. On another occasion, a user representative praised the capability for live

[6] Recall the mnemonic edifices constructed by classical orators (Yates, 1969) to assist the delivery of a speech in oral cultures. The spatial distribution of their mental images superimposed on familiar settings waymarked the route of the speech. Various authors in the social sciences have observed that experienced time is socially configured and patterned (e.g. McGrath and Kelly, 1986; Urry, 1996).

modifications at a feedback session. The full participation of users also meant that they could be involved in some development work. This was possible in an intranet project because users could realistically undertake HTML authoring. Clearly, this may not generalise to developments where the skills required are more specialised. Preparation of training material was another user activity. A training document mentioned that a lack of formal training in the previous iteration had led to some ignorance of its full capabilities leading to some (internal) customer resistance. Accordingly, end-user training was emphasised during our development. An important activity for the user representatives was preparation of training material for 'supertrainers' who would coach users in their region. In sum, the role of user involvement in the development was to contribute knowledge of end-user practice, provide immediate feedback on evolving prototypes to the developers, undertake some HTML authoring, participate in design work using low technology mock-ups and develop training material. In the wider context of our research, it was clear that developers valued user involvement because they saw a potential connection with increased user ownership and advocacy of the resulting system (Tudhope et al., 2000). As a construction company development manager put it:

> It also means that if there's six people around the table, you know there'll be six people at the end who'll want to have a go at it and will be keen to push it. Whereas any system that's owned almost purely by IT and IS, you then have to install it and you have to gain this acceptance of systems. All our past bad systems have been done really by ourselves with virtually no user involvement.

While the case study illustrates potential benefits of intensive projects and the clean room, there were also problematic aspects. In this project it did not prove possible to insulate user representatives from their other duties. They not only had to do their duties on the project, but also had to field e-mails and telephone calls from colleagues back at their home offices, which usually took up the first hour or so of each day. This put some strain on the users — one was asked what he was doing that evening, and replied, somewhat laconically, "my day job." He pointed to a number of files under his arm, saying "I've got to get through all that before ten tomorrow morning, so its tea in the room for me." Other factors are the expense involved for the organisation, both in monetary terms and in terms of impact on users' routine tasks, and the requirement for team members to be absent from home during the week days of the project. Intensive projects are obviously easier to achieve when they are in-house, as opposed to external contracts. However, expense and user time are also likely to be issues in the three month variant of RAD. The RAD project briefly discussed in Section 5.1 offers a halfway house, in that it was intensive for developers but not users.

Thus, it would appear that intensive projects are useful in some situations but careful consideration needs to be given to resource implications. Grudin (1993) has discussed general obstacles to user involvement in large organisations. The need for a 'culture change' emerged in both the RAD literature (DSDM Consortium, 1995) and in conversations with RAD developers in the case study company — in part this entails a greater commitment by the customer to participation in prototyping projects and associated workshops. There are implications here for training and education in systems development and procurement. On one of our visits to the company, we asked if they had considered the use of electronic environments to enable remote team members to have virtual meetings or even shared screen demos. It was not then being seriously considered, although it was acknowledged that intensive RAD developments were expensive in travel costs. The reasons given were the advantage of having everyone together and that (say) 18 people were too many for video conferencing techniques. We also inferred that possible reasons might include the practical implications at that time for a large organisation with many centres and a perception that group dynamics at meetings might be affected (motivation of user representatives at project meetings had been emphasised as a concern). As the necessary broadband infrastructure becomes more widespread, distributed collaborative systems might offer the potential for maintaining virtual project memory walls (see for example Roseman and Greenberg, 1996; and for a review of earlier work including ‘video walls’, Root, 1988). However, the resulting implications for project management, work practice and user involvement require further investigation.

5.4. Future work

Studying an intensive design project has highlighted issues of design representation and time management and how they might serve to afford and motivate user involvement. Further ethnographic studies are needed of non-intensive RAD projects over more extended timescales.

Acknowledgements

This project was funded by the UK Economic and Social Research Council (Grant No. R000 23 5505). We would like to thank all those who talked to us about their work, and in particular the participants in the project discussed. We would also like to thank Chris Carne, Tia DeNora, Keith Verheyden and Clare-Marie Karat and the anonymous reviewers for their comments and suggestions.

References

Alavi, M., 1984. An assessment of the prototyping approach to information systems development. Communications of the ACM 27 (6), 556–563.

Anderson, R., 1994. Representations and requirements: the value of ethnography in systems design. Human Computer Interaction 9, 151–182.

Bates, P., 1995. Rapid application development — concept, methodology, or what?. In: Stephens, M. (Ed.). Proceedings of the Seminar Series on New Directions in Software Development: RAD — How Rapid is Rapid?. University of Wolverhampton, UK, pp. 1–21.

Beynon-Davies, P., Tudhope, D., Mackay, H., 1999. Information systems prototyping in practice. Journal of Information Technology 14, 107–120.

Beynon-Davies, P., Carne, C., Mackay, H., Tudhope, D., 1999. Rapid application development (RAD): an empirical review. European Journal of Information Systems 8, 211–223.

Blomberg, J., Giacomi, J., Mosher, A., Swenton-Hall, P., 1993. Ethnographic field methods and their relation to design. In: Schuler, D., Namioka, A. (Eds.). Participatory Design: Principles and Practices. Lawrence Erlbaum, Hillsdale, NJ, pp. 123–155.

Bødker, S., 1991. Through the Interface: a Human Activity Approach to User Interface Design. Lawrence Erlbaum, Hillsdale, NJ.

Bødker, S., Grønbæk, K., 1991. Cooperative prototyping: users and developers in mutual activity. International Journal of Man–Machine Studies 34, 453–478.

Button, G., Sharrock, W., 1993. Practices in the work of ordering software development. In: Firth, A. (Ed.). The Discourse of Negotiation. Pergamon, Oxford, pp. 159–180.

Carey, T., Mason, R., 1983. Information systems prototyping: techniques, tools, and methodologies. INFOR — Canadian Journal of Operational Research and Information Processing 21 (3), 177–191.

Carmel, E., Whitaker, R., George, J., 1993. PD and joint application design: a transatlantic comparison. Communications of the ACM 36 (4), 40–48.

DSDM Consortium, 1995. Dynamic Systems Development Method. Tesseract, Farnham, UK (vol. 2).

Egger, E., Wagner, I., 1993. Negotiating temporal orders: the case of collaborative time-management in a surgery clinic. Computer Supported Cooperative Work 1 (4), 255–275.

Ehn, P., Kyng, M., 1991. Cardboard computers: mocking it up or hands-on the future. In: Greenbaum, J., Kyng, M. (Eds.). Design at Work: Cooperative Design of Computer Systems. Lawrence Erlbaum, Hillsdale, NJ, pp. 169–196.

Fabre, J., Howard, S., 1998. Introduction to the special issue on temporal aspects of usability. Interacting with Computers 11 (1), 1–7.

Gordon, V., Bieman, J., 1995. Rapid prototyping: lessons learned. IEEE Software 30 (1), 85–95.

Gould, J., Lewis, C., 1985. Designing for usability: key principles and what designers think. Communications of the ACM 28 (3), 300–311.

Graham, I., 1989. Incremental development: review of non-monolithic life-cycle development models. Information and Software Technology 31 (1), 7–20.

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

Grudin, J., 1993. Obstacles to participatory design in large product development organisations. In: Schuller, D., Namioka, A. (Eds.). Participatory Design: Principles and Practices. Lawrence Erlbaum, Hillsdale, NJ, pp. 99–122.

Harker, S., 1991. Requirements specification and the role of prototyping in current practice. In: Karat, J. (Ed.). Taking Software Design Seriously: Practical Techniques for Human–Computer Interaction. Academic Press, London, pp. 339–354.

Hartson, H., Smith, E., 1991. Rapid prototyping in human–computer interface development. Interacting with Computers 3 (1), 51–91.

Heath, C., Luff, P., 1992. Collaboration and control: crisis management and multimedia technology in London underground line control rooms. Computer Supported Cooperative Work 1 (1), 69–94.

Hughes, J., Randall, D., Shapiro, D., 1992. Faltering from ethnography to design. Proceedings ACM Conference on Computer Supported Cooperative Work, pp. 115–122.

Hughes, J., Sommerville, I., Bentley, R., Randall, D., 1993. Designing with ethnography: making work visible. Interacting with Computers 5 (2), 239–253.

Karat, J., Bennett, J., 1990. Supporting effective and efficient design meetings. Proceedings Conference on Human–Computer Interaction — Interact'90. Elsevier Science (North-Holland), Amsterdam, pp. 365–370.

Karat, J., Bennett, J., 1991. Using scenarios in design meetings — a case study example. In: Karat, J. (Ed.). Taking Software Design Seriously: Practical Techniques for Human–Computer Interaction. Academic Press, London, pp. 63–94.

Karat, J., Dayton, T., 1995. Practical education for improving software usability. Proceedings ACM Conference on Computer Human Interaction, pp. 162–169.

Lilienthal, C., Zullighoven, H., 1996. Techniques and tools for continuous user participation. Proceedings Participatory Design Conference, pp. 153–159 (Computer Professionals for Social Responsibility).

Martin, J., 1991. Rapid Application Development. Macmillan, New York.

McGrath, J., 1990. Time matters in groups. In: Galegher, J., Kraut, R., Egido, C. (Eds.). Intellectual Teamwork, Social and Technological Foundations of Cooperative Work. Lawrence Erlbaum, Hillsdale, NJ, pp. 23–61.

McGrath, J., Kelly, J., 1986. Time and Human Interaction: Toward a Social Psychology of Time. The Guildford Press, NY.

Mogensen, P., 1992. Towards a provotyping approach in systems development. Scandinavian Journal of Information Systems 4, 31–53.

Muller, M., 1991. PICTIVE — an exploration in particpatory design. Proceedings ACM Conference on Computer Human Interaction, pp. 225–231.

Muller, M., 1993. PICTIVE: democratizing the dynamics of the design session. In: Schuler, D., Namioka, A. (Eds.). Participatory Design: Principles and Practices. Lawrence Erlbaum, Hillsdale, NJ, pp. 211–237.

Root, R., 1988. Design of a multi-media vehicle for social browsing. Proceedings ACM Conference on Computer Supported Cooperative Work, pp. 25–38.

Roseman, M., Greenberg, S., 1996. TeamRooms: network places for collaboration. Proceedings ACM Conference on Computer Supported Cooperative Work, pp. 325–333.

Stapleton, J., 1997. Dynamic Systems Development Method: The Method in Practice. Addison-Wesley, Harlow.

Suchman, L., 1987. Plans and Situated Actions: The Problem of Human Machine Communication. Cambridge University Press, Cambridge.

Suchman, L., 1992. Technologies of accountability: of lizards and aeroplanes. In: Button, G. (Ed.). Technology in Working Order: Studies of Work, Interaction, and Technology. Routledge, London, pp. 113–126.

Suchman, L., Trigg, R., 1993. Artificial intelligence as craftwork. In: Lave, J., Chaiklen, S. (Eds.). Understanding Practice. Cambridge University Press, Cambridge, pp. 144–178.

Tudhope, D., Beynon-Davies, P., Mackay, H., 2000. Prototyping praxis: constructing computer systems and building belief. Human–Computer Interaction 15 (4) In Press.

Urry, J., 1996. How societies remember the past. In: Macdonald, S., Fyfe, G. (Eds.). Theorizing Museums. Blackwell, Oxford, pp. 45–65.

Wall, P., Mosher, A., 1994. Representations of work: bringing designers and users together. Proceedings Participatory Design Conference, pp. 87–98 (Computer Professionals for Social Responsibility).

Wood, J., Silver, D., 1989. Joint Application Design: How to Design Quality Systems in 40% Less Time. John Wiley, New York.

Woolgar, S., 1991. Configuring the user: the case of usability trials. In: Law, J. (Ed.). A Sociology of Monsters. Routledge, London, pp. 58–100 (an updated version appears in: Grint, K., Woolgar, S., 1997. The Machine at Work: Technology, Work and Organization. Polity, Cambridge, pp. 65–94..

Yates, F., 1969. The Art of Memory. Penguin Books, London.