Cover page:

A case study on integrating contextual information with analytical usability evaluation

Corresponding author:

A. Blandford
UCL Interaction Centre
University College London
26 Bedford Way
London WC1H 0AB
U.K.

A.Blandford@ucl.ac.uk

Preprint: final version available from

BLANDFORD, A. & RUGG, G. (2002) A case study on integrating contextual information with usability evaluation. *International Journal of Human-Computer Studies*. 57.1, 75-99.

A case study on integrating contextual information with analytical usability evaluation

A. Blandford & G. Rugg
UCL Interaction Centre
University College London
26 Bedford Way
London WC1H 0AB
U.K.
A.Blandford@ucl.ac.uk
http://www.uclic.ucl.ac.uk/annb/

Department of Computer Science
University of Keele
Staffordshire
ST5 5BG
U.K.

Abstract

The work reported here integrates an analytical evaluation technique, Programmable User Modelling, with established knowledge elicitation techniques; the choice of techniques is guided by a selection framework, ACRE. The study was conducted in conjunction with an ongoing industrial design project. Techniques were selected to obtain domain knowledge in a systematic way; the rationale behind each choice is discussed. The use of 'negative scenarios' as a means of assessing the severity of usability findings is introduced.

Keywords: PUM, usability evaluation, ACRE, knowledge elicitation, design

Introduction

One of the greatest challenges for HCI is arguably integrating analytical usability techniques in a usable and effective way with each other and with design practice, working with inevitable commercial and resource constraints. The work reported here originated in a study of how one particular analytical approach, Programmable User Modelling (PUM: Young, Green & Simon, 1989; Blandford & Young, 1996) could be more effectively integrated with design practice. However, the focus of this paper is not on the PUM approach in particular, but on the experience of integrating an analytical evaluation technique with a range of knowledge elicitation techniques, within an industrial design project.

Although there have been demonstrations in laboratory or controlled settings that individual analytical evaluation techniques can make a useful contribution to design (e.g. Gray, John & Atwood, 1993), and of how different techniques can be integrated with each other (e.g. Bellotti, Blandford, Duke, MacLean, May & Nigay. 1996), relatively little work has been reported that directly addresses the issues involved in integrating analytical techniques with the (often messy) realities of design practice. One example is the work of Spencer (2000). There are normative descriptions (e.g. Beyer & Holtblatt, 1998; Mayhew, 1999) that propose approaches to design that start from a 'clean slate',

proposing a particular design process and set of techniques that fit within that process. In parallel, there are descriptions of design case studies in which researchers have contributed to design by being participants in the design team (e.g. O'Neill, 2000; Coble, Karat & Kahn, 1997; Good & Blandford, 1999).

These lines of research contribute to an overall big picture of how a repertoire of usability techniques, both empirical and analytical, can be integrated with each other and with ongoing design practice. The work reported here contributes another small piece to the big picture. That is: we are concerned with understanding existing design practice, and investigating how a set of appropriate techniques can be integrated, both with each other and with that design process. Research in this area inevitably has to accommodate the constraints of that design practice: as has been observed (e.g. Bellotti, 1989; Buckingham Shum & Hammond, 1994), the gulfs between HCI research and design practice are substantial, and difficult to overcome. In the study reported here, the investigators worked alongside an ongoing design process, working within many of the constraints imposed by that process, but not fully integrated with it due to resource constraints. One important feature of the design process was that it could be characterised as 'evolutionary': the new design was extensively based on an existing, well tested and understood, design. Another was that the design team were software engineers, with no background in usability. The focus of this work is on how established techniques can be applied selectively and sequentially, to contribute user-oriented findings into the design process.

Background

Usability evaluation techniques are designed to support an analyst in assessing the usability of a device or identifying potential usability problems. While much evaluation is empirical, typically testing a device with a representative sample of users, an alternative is analytical evaluation. Analytical techniques can complement empirical evaluation: empirical approaches identify what users do, but analytical techniques can help ascertain why users have particular difficulties or how the device might be redesigned to overcome those difficulties (Hollnagel, 1993).

Numerous analytical evaluation techniques exist, varying considerably in sophistication and formalisation (Dix, Finlay, Abowd & Beale, 1998; Newman & Lamming, 1995). For example, heuristic evaluation (Nielsen, 1994) is a low-cost approach that is based on assessing an interface against a check-list; this can yield useful results for interface design questions such as: 'is the interface consistent?' or 'are the labels easy to understand?'. However, as Connell and Hammond (1999) observe, heuristics are sufficient for identifying surface difficulties, but more theoretically grounded usability principles are needed to identify deeper difficulties with a design. At the other end of the spectrum, techniques such as GOMS (Card, Moran & Newell, 1983; John & Kieras, 1996), require very detailed and complete analysis, and yield precise predictive results – for example, assessing how long a particular task will take an expert user. A common feature of most of these techniques is that they focus on the device out of context; that is: they assess the

usability of the device taking the domain (and its representation through the device) as 'given'.

Conversely, there is now a growing body of techniques that are directly concerned with understanding use in context as a basis for design. One of the more widely known is Contextual Design (Beyer & Holtzblatt, 1998), which outlines an approach to observing users in the workplace, describing their activities and use of artefacts, and developing a set of models that are used as a basis for design. Such techniques appear to be based on the assumptions that the new design is not constrained to be an adaptation of an existing product and that the whole design team have the appropriate skills in observation and user- and domain-centred analysis. In many design projects, including the case study described here, neither of these assumptions hold. The approach developed here takes the existing design as a starting point, and introduces usability analysis alongside the software engineering process.

In this section we discuss understanding use in context, focusing on understanding how knowledge is used in context, then present a short introduction to Programmable User Modelling. In the following section, we present the case study. We close with a discussion of issues raised by this work.

Understanding the device in context

As noted above, most established analytical evaluation techniques either work without any domain knowledge or leave that knowledge implicit (relying on the analyst's pre-existing understanding of the domain). Indeed, it is difficult to find examples that integrate domain knowledge with usability evaluation, although the need is discussed (e.g. Hollnagel, 1998). One obvious solution is to recruit the power of knowledge elicitation techniques to address this limitation. The term 'knowledge elicitation' is used here in the broad sense of eliciting knowledge from a human being for any purpose, rather than the narrower sense of being a stage of knowledge-based systems development.

A broad range of knowledge elicitation techniques have been developed, including ethnographic approaches (e.g. Sommerville, Rodden, Sawyer, Bentley & Twidale, 1993), scenarios (e.g. Gough, Fodemski, Higgins & Ray, 1995; Carroll & Rosson, 1992; Rosson & Carroll, 2002) critical incidents (Flanagan, 1954), and more formalised approaches deriving from Personal Construct Theory (Kelly, 1955). Each of these techniques is suited to particular situations, and delivers particular kinds of results.

The issue of technique selection is a well recognised problem. Some guidance is provided by quantitative comparisons of techniques (e.g. Burton, Shadbolt, Rugg & Hedgecock, 1990) and by other pragmatic factors in support tools such as KEW (Shadbolt & Wielinga, 1990) and AQUINAS, (e.g. Boose, Shema & Bradshaw, 1989). These types of guidance, however, do not address the question of whether some techniques might systematically miss some types of knowledge which could be elicited via other techniques. This question has received considerable attention in social science research, and a widely used approach from that area is triangulation, in which several techniques are

used to cross-check and cross-validate each other (e.g. Denzin & Lincoln, 1994). This, however, does not provide any theoretically-grounded basis for the choice of techniques, and does not provide any guarantee that the chosen techniques will between them elicit the relevant information.

Thus, as well as techniques, there is a need for a framework to guide selection and sequencing of the individual techniques. The framework used in this study was the ACRE framework (Maiden & Rugg, 1996). This is a faceted framework which includes categorisation of techniques in relation to types of knowledge and types of communication, drawing on the established psychological literature on these topics. This theoretical grounding makes it possible to make detailed, specific predictions about which techniques are suitable or unsuitable for eliciting particular types of knowledge; it also provides theoretically grounded guidance on identifying the different types of knowledge, as illustrated below in relation to compiled skills.

The knowledge and communication types are divided into three main categories and further subcategories. The main categories are:

- explicit knowledge (knowledge which is readily available to introspection, and accessible by any elicitation technique - for instance, familiar personal details such as ones own name, place of birth, *etc*.);
- semi-tacit knowledge (knowledge that can be accessed by some techniques but not by others); and
- tacit knowledge (knowledge which is not accessible to introspection via any elicitation technique).

There are various types of semi-tacit knowledge, including:

- taken for granted knowledge, which will not be explicitly mentioned in verbal interaction (Grice, 1975): taken for granted knowledge involves information which is so familiar to the respondents that they do not bother to mention it explicitly, since they (often incorrectly) assume that everyone else also knows it. For instance, a trainer might take it for granted that trainees with personal computer experience have obtained this on a PC rather than a Mac. Several examples of taken for granted knowledge occurred during this study, and are described below. The very ubiquity which causes information to be taken for granted also means it is likely to be important.
- front and back versions, i.e. the 'official' account of what should happen, as compared to the more realistic account of what actually happens (Goffman, 1959). For instance, companies typically play down the incidence of security problems when dealing with the general public, but will give a more accurate account to specialists working for them on such problems.

- knowledge that depends on recognition as opposed to recall. For instance, most Word users can recognise all the standard menu bar items, but would have difficulty recalling them.

Tacit knowledge is subdivided into:

- compiled skills – skills which were once explicit or semi-tacit but which have since become so habitualised as to become inaccessible to introspection. E.g., touch typists can perform touch typing at a high level of accuracy, but typically have difficulty answering questions such as "which key is to the left of 'g'?"
- implicit learning – learning which occurs without any explicit or semi-tacit stage, and is therefore tacit throughout, making it inaccessible to introspection (Seger, 1994). E.g., an experienced driver will usually know how their car sounds and feels when it is operating properly, but will usually be unable to articulate this knowledge.

The ACRE framework explicitly highlights the ways in which knowledge and communication types constrain the information which can be elicited via a particular elicitation technique. For example, traditional interviews are likely to miss a significant proportion of semi-tacit knowledge; conversely, interviews can quickly clarify some issues and raise others that need to be investigated by some other approach. A similar framework is proposed by Robertson (2001), who provides brief descriptions of various techniques for discovering requirements, and categorises them according to whether they are useful for eliciting requirements that are 'conscious' (i.e. explicit knowledge), 'unconscious' (tacit and semi-tacit) or 'undreamed' (future possibilities that have not so far been considered at all); in the work reported in this paper, we do not consider 'undreamed' requirements, but focus on evolutionary design from an existing artefact, and on the order in which techniques are to be applied.

An introduction to PUM

This study integrates knowledge acquisition with analytical evaluation. As noted above, the particular approach to evaluation used in this study is Programmable User Modelling (Young, Green & Simon, 1989). Traditional PUM (e.g. Blandford & Young, 1996; Blandford, Buckingham Shum & Young, 1998) is a formal analysis technique that shares features in common with both GOMS (Card, Moran & Newell, 1983) and Cognitive Walkthrough (Wharton, Rieman, Lewis & Polson, 1994). All three approaches are concerned with goals and knowledge. Whereas GOMS specifies expert knowledge of action sequences, and Cognitive Walkthrough focuses on whether the user can discover what to do next based on information from the interface (and therefore focuses mainly on novice interactions), PUM starts from a specification of knowledge and uses that as a basis for inferring what rational user behaviours are possible or likely.

Initially, in this study, a lightweight version of the approach called 'PUMA footprints' (Blandford, Butterworth & Curzon, 2001) was applied; this is an inspection technique (Neilsen and Mack, 1994) that involves the analyst considering possible user behaviours and identifying points in the interaction where the user might adopt incorrectly

formulated goals, or have incorrect knowledge of the state of the device. The features of a design that might provoke such errors are referred to as 'footprints' of the design. Looking ahead to the case study (below), three of the 'footprints' are particularly relevant:

- ***Post-completion errors***: When the user has achieved their main goal, they are liable to terminate the interaction without addressing any outstanding sub-goals. This problem has been thoroughly analysed by Byrne and Bovair (1997), who demonstrate that, while it is an intermittent problem, it is persistent, cannot be eliminated through improved training, and does not disappear as users become experts. The footprint of a post-completion error is that "there is a precondition to the conceptual operation that achieves the main goal, but satisfying the precondition perturbs the state, and a clean-up action is needed after achievement of the main goal".
- ***Lack of observability***: When the user needs to know particular things about the state of the device to assess its state relative to their goals, but the device does not display sufficient information about its current state for the user to know all they need to, the device lacks observability.
- ***Lack of predictability***: When the user has a goal (that the device supports) and knows of actions that are likely to achieve that goal, but the user cannot use their knowledge of the current state and of the effects of actions to predict the effect of their next action, the device lacks predictability.

Once areas of concern have been identified by applying this lightweight approach, more formal analysis can be conducted to develop a deeper understanding of issues.

The case study

As discussed above, the study reported here aimed to integrate knowledge about the context of use with analytical evaluation. In particular, we focused on the systematic use of knowledge elicitation techniques with PUM to ensure that domain knowledge was integrated as necessary with usability considerations. The aim was to test our hypothesis that these various techniques could usefully be integrated within a process that would give good leverage on evolutionary design. The method applied was based on a case study, conducted in collaboration with Domino Printing (Domino, 2002).

The company

Domino manufactures industrial ink-jet printers. Their products are used for printing sell-by dates on packets of food, cans of drink, etc., and have world-wide sales. Printers are typically located on production lines, where they print a label on each product item (e.g. packet or tin) as it goes past. Each ink-jet machine has its own control system, generally located next to it in the working environment. Operators access the control system to 'program' the codes (or labels) that are to be printed (e.g. "best before 27 Jun"), to adjust the time between when a product is detected and when the printer activates (e.g. each

packet is detected at a particular point on the production line and the code is printed on in 4.3 seconds later as it passes under the print-head), etc.

The focus of the company's work is on developing suitable inks, printer hardware and controller software, so that printing is consistently accurate and of high quality. The bulk of their engineering effort goes into ensuring that the system works reliably. The design team has a high level of expertise in software engineering and in the application domain (printing technology), but less in human factors or organisational issues.

Domino do not design products to a particular customer's specification, but aim to anticipate customers' requirements and produce a marketable product. One of the natural consequences of this is that revisions of the product can lead to "creeping featurism". Domino staff believed that a shift of emphasis from "more features" to "better usability" would improve marketability, and were therefore open to suggestions for design modifications (though the momentum of the ongoing design process was such that it would probably take some time for any suggestions to be acted upon). Because the design team had not established access to end-users for explicit requirements acquisition or usability testing, these activities took place informally – through feedback from the customers via the sales team, and by testing products on other employees in the company.

The product

The case study involved a new product, which was in the late stages of design when the study started. At the time of initial contact, some aspects of the prototype design were already determined, while others were still subject to possible revision. The design of the new system was based largely on that of the previous system, and embodied the design team's intuitions about what makes a usable interface; the designers' thinking about the new design was strongly influenced by the existing, well understood, design. Interaction requirements were not explicitly represented in design documentation, but were simply embodied in the current prototype.

The interface

The existing interface centred on an LCD panel, which supported menu navigation via soft keys, and also included a QWERTY keyboard. Some of the menus included deep tree structures.

The prototype interface for the new system was similar, but included a new pad of eight function keys. These function keys include a lockout key (as described below), a service key (which permits access to service functions that are only used by maintenance experts), and six keys that allow the operator to adjust machine set-up parameters and to design and retrieve print codes. For reasons of commercial confidentiality it was not possible to take away pictures of the proposed interface, but similar interfaces can be viewed on the company's web site (Domino, 2002).

Method

In preliminary discussions with the company, three usability issues were highlighted as possibilities for study: the layout of buttons; the appropriateness of the menu structures; and the use of the security feature. Of these, the first two were more amenable to empirical approaches to evaluation: layout would ideally be investigated by presenting users with alternative layouts, and menu structures required detailed investigation of common and important task structures in end-user organisations. Also, in the initial discussion, the problem of post-completion errors in the design of the security feature was identified, so this was selected as the focus for the study.

A four-stage method was devised. This was designed to elicit knowledge from appropriate people and conduct analysis in a sequence that gave maximum information with limited resources:

1. Initial knowledge elicitation, at the company's site, to familiarise the investigators with the current design, determine context of use, elicit requirements, familiarise with the proposed design and establish the views of the company representatives on particular issues.
2. Analytical evaluation of the proposed design. The evaluation in turn identified possible problem areas whose seriousness could only be determined by understanding the context of use.
3. Investigation of the domain, using knowledge elicitation techniques to clarify the context of use with relation to the potential problems. This resulted in several suggestions for re-design.
4. A report, summarising the findings and results of analysis and making various recommendations, was submitted to the company, and their feedback on the report and its recommendations was received.

The security feature in the existing system required an authorised user to enter the password to access certain facilities, complete their work, then reset password protection before leaving the machine. This was to be implemented in a similar way in the new design. This description matches the footprint for a post-completion error.

The design team was aware that the security feature was not widely used, but did not understand why. While the proneness of the design to post-completion errors was one possible source of the problem, it was likely that there were other difficulties that were not immediately apparent. In addition, the consequences of post-completion errors would depend on the context of use. We believed that focusing on the design of the security mechanism would yield a better understanding of use (and reasons for non-use) of the feature, and also help towards proposing a re-design.

Stage 1: initial investigation (requirements acquisition based on company information)

Procedures and questions for the initial knowledge elicitation phase were devised to explore general design issues, and to probe particularly into security and the design of the security mechanism. This was necessary to familiarise the investigators (as 'outsiders') with the design problem.

The data gathering began with unstructured interviews conducted with two members of staff from stakeholder groups within the company. This was followed by a demonstration of the current product and an introduction to the new system.

Interviews

Two separate, single-respondent interviews were conducted. One interviewee was a software designer; the other was responsible for training users of the current product, and for designing training materials for the product under development. Respondents were interviewed separately so that their responses could be compared – both for points of commonality and also for issues on which their views diverged. These interviews took place in a meeting room where the interviewees would not be distracted by the normal demands of their jobs; each lasted about 30 minutes. Interviews were taped; transcriptions were analysed with a particular focus on how the interviewees believed the product would be used, on the design of the security mechanism, and on any points where the views of the two interviewees diverged. In the extracts from transcripts presented below, '[…]' is included to indicate that a portion of the transcript has been omitted for the sake of brevity.

The interviews were performed before other elicitation techniques so that gaps in coverage, due to taken for granted knowledge etc., could be identified. If the demonstrations had occurred first, then it would have been harder to identify any gaps in coverage, since the investigators would share at least some of the taken for granted knowledge as a result of episodic memory (Tulving, 1983), even if this knowledge had never been explicitly mentioned anywhere in the demonstration.

Here, we summarise the main findings from the interviews to illustrate the types of data acquired by this method and the way in which issues for further investigation emerged.

Interview findings

Many important aspects of the design in general were described by the designer interviewed, when he was asked to describe the most frequent and most important tasks:

> "Something like 85% of users make use of a very limited set of facilities, and the facilities that are used most would, the sequence of events would be they would switch the machine on, so switch the mains power on, they would press one button that sequences the ink system and print head on, so that gets the ink system pressurised and gets the print head to do what it needs to do to control and be able to print, and while that sequence on is happening, the user would type into the WYSIWYG editor the text or the sell-by date or whatever they want to print and hit the 'print message' button, which would cause that label then to be printed. And then at some later time when they've finished printing, at the end of

> the production run or whatever, they would perhaps press one button again to sequence the ink system off, which would cause the printer to shut down and clean itself out, flush all the ink out of the print head and everything."

We see here the designer spontaneously presenting his view of a typical scenario of use. General scenarios, however, do not help with understanding when and how the security feature might be used. For this, critical incidents were useful, as these provide examples of cases where the feature failed in some way. For example, the trainer told a story:

> "This was a long long time ago, but we had a guy who had been subject to disciplinary procedures in a brewery, and it was in the run-up to Christmas, and he finally over-stepped the mark and was told to leave site there and then, and as he did he went through the canning hall and he put a fairly short, sharp message, which wasn't exactly a Christmas greeting, and they managed to run, I think they managed to run four or five hours worth and because it was the run-up to Christmas they could not afford to put that product to one side [...] so they got a whole lot of contract cleaners in."

From the interviews, it was clear that the design team was aware of design questions about some features of the product, and had debated them at length, and that they made use of critical incidents from related industry sectors to inform the debate, particularly when considering the security feature.

In response to such security violations, password protection had now been implemented for various levels of access to the device. When discussing security, the designer talked in terms that indicated that he viewed the password feature of the system as the solution to the security problem. In contrast, the trainer focused more on procedures and working practices:

> "They will do the shift change and somebody will come along and will actually check the code and the line supervisor will sign their sheet and attach the board to the thing with the samples in and they'll then take that up to the production manager and maybe to the quality assurance manager who will also sign it off. You tend to find the factories that will do that are the ones that have had the bad experience and run sixteen hours of production with the wrong code."

One significant difference between the two interviews (designer and training manager) involved how often the respondents thought that typical users would wish to re-program the device. If the device is only re-programmed occasionally, and is password-protected, the likelihood of passwords being lost or forgotten by legitimate users is high (Adams & Sasse, 1999).

The designer believed that users would typically re-program the device daily. When asked about types of users, he explained:

> "The assumption we made is that users have no knowledge of the machine, or at best very limited knowledge, and they're not necessarily frequent users. They wouldn't be standing at the machine all day, for instance. They'd maybe make some adjustments once per day, so they're not going to spend a lot of time at it."

The trainer, in contrast, thought of users as a spectrum:

> "The way the customer works with it, means that depending on the kind of code that they're using, because some people use codes that are like an incremental number, some people just put maybe ingredients onto a box which could stay the same; other people put things like real time information in, which is going to be constantly changing, some people use codes that change once a day and they actually go to the machine and type them in once a day. Other customers will put a message in and then as long as there are no failures or anything then the machine will just run with that setting for the next six months or whatever. [...] so you've got people who have to be able to use the interface lots and lots of times a day down to the extreme where it's a really high pressure situation where there's maybe been a problem and they're trying to reinstall the codes and it could be six months since they last did it."

Both designer and trainer (implicitly) considered the user to be any person who adjusts the machine: if the machine is adjusted once a day, then it is used once a day; the question of whether the user is one or several individuals (who each might therefore use the machine much less frequently) was not addressed. These issues of frequency and type of use were addressed through other techniques later in the investigation.

In their interview accounts of how the machine would be set up and used, neither designer nor trainer mentioned having to enter a password. A question then was whether this was taken for granted knowledge – as was apparent in the designer's high-level description of other aspects of using the device – or whether they actually believed that most users disable the security feature. This point was investigated later: discussions with the sales representative, and with the designer in the final debriefing, made it clear that the current assumption was that the security mechanism is not widely used, but that design revisions would cause that to change.

Demonstration

Following the interviews, the same two members of the company gave a demonstration of the current production model. The demonstration was used as an opportunity both for observation (by the investigators) and for on-line self-report (by the company representatives) in a context which would be familiar and comfortable for the respondents. The observation was used as a means of identifying tacit knowledge and some types of semi-tacit knowledge; in particular, the on-line self-report aspect was used for probing semi-tacit knowledge about the current design.

In the demonstration, the investigators were shown how all user functions are accessed via a hierarchical menu. The demonstration was video-taped for subsequent analysis. The following is an extract from the discussion that took place while the trainer (M) and software engineer (S) were demonstrating the system to the investigators (one of whom is indicated as "I"). This illustrates the greater level of detail, and the increased focus, that is possible with on-line self-report: a level of detail that the interviews could not capture.

> M: Normally when you're demonstrating the machine you leave that off, because if you hit the escape too many times it's really annoying; you'd have to enter the password far more often than you'd ever want to. But if you're on the production line where you don't want people to change codes or select other messages or

whatever, then it's a really useful: lockout to actually protect the code that you're applying.

I: So just hitting escape enough times on this version is sufficient to take you

S: takes you back up the tree.

M: It pops you out to the top, and unless you know the code you can't get in to do anything else, to change, I'll switch it off, if that's OK, because it becomes quite annoying if you're moving around the software trying to do other things [presses >>, >>, 4th soft-key,3rd soft-key rapidly]

I: So that's why you said lots of people leave it disabled?

S: That's right, yes. That's certainly the case with this machine. It should be less of a problem with the new one because there's no concept of having to escape multiple times to get back to the top of some tree. [presses 2nd, right-cursor, 3rd, esc, esc, esc, <<, << rapidly]

As well as providing more detail, the demonstration also confirmed information gained in interviews. For example, the fact that the lockout feature was recognised as being difficult to use was illustrated by the fact that after a few minutes the trainer disabled it, because it was too easy to accidentally activate it.

Equally importantly, the demonstration of the existing design gave the interviewees a focus for discussing the proposed design changes by comparing the new design with the existing product. For example, as shown in the extract above, the design team could discuss their assumption that the revised design would make lockout "less of a problem".

Then followed an introduction to the new system, which involved presenting the prototype interface and talking through the design. Again, the new design was described largely by comparison with the current design.

These demonstrations made it possible to expose semi-tacit knowledge, and to validate information acquired through the earlier interviews.

Stage 2: Usability evaluation

Once the investigators had a reasonable understanding of the design and its context of use, it was possible to conduct usability evaluations – first using PUMA footprints as described above, then more formal PUM analysis (Blandford, Buckingham Shum and Young, 1998). This approach was used because one of the initial objectives was to test real-world application of PUM. In addition: it would be difficult to test in-context use of a feature that is hardly used in practice; it is very hard to provoke post-completion errors under controlled conditions (Byrne and Bovair, 1997); and Domino had limited access to end users, so it would be difficult to set up substantial user trials.

Having only that knowledge of the domain which had been acquired through interviews and demonstration, we had to start with the interface and interaction as presented, and ask further questions about the domain and context of use later. Therefore, the initial focus was on understanding both the current design and the proposed re-design. In effect,

one of the roles of analysis was to help the analysts improve their understanding of the design prior to any empirical study.

Inspection using PUMA footprints

We start with a brief description of the new system to show how an inspection technique can highlight important usability issues.

In the new system, lockout is enabled or disabled from a dialogue screen within the machine set-up area; it is activated by pressing a dedicated button on the keyboard. When the user presses the "lockout" button, the lockout screen is displayed; if lockout is enabled then other areas cannot be accessed until a password has been entered. However, if lockout is disabled, other areas (except the service area) can still be accessed from this point. Anyone entering the service level password has access to all areas.

Referring to the footprint for observability, we see that the device does not display the current settings of all state components (whether or not lockout is activated) to identify when a goal has been achieved (whether the security mechanism has actually been activated). This aspect of the state is not observable.

In addition, referring to the footprint for predictability, the user may not have sufficient knowledge of the current state of the device (whether lockout is enabled) to appropriately predict the effect of an action (whether simply pressing lockout will activate the security mechanism). Therefore, the device is also not predictable.

As Rosson and Carroll (2002) observe, as well as identifying possible difficulties, it is important to assess their severity (how often they are likely to occur or how serious the problem if they do occur). As a starting point for this, we developed negative scenarios.

Negative evaluative scenarios

Starting from the results of the usability inspection, scenarios of use in which usability difficulties would arise can be identified. In this particular case, the user will be unable to predict the effect of pressing lockout under various conditions:

a) If the context of use is one in which lockout is sometimes enabled and sometimes disabled, the user will have no way of telling whether or not lockout is currently enabled, and will not know whether or not the system is actually in lockout mode when the lockout screen is displayed.

b) If the context of use is one in which lockout is usually enabled, the user will believe that the system is locked out whenever the lockout display is showing. However, if lockout has actually been disabled, the system will be left accessible to unauthorised users.

c) If the context of use is one in which lockout is usually disabled, but it has been enabled (accidentally or maliciously), a service level user may press lockout to exit from service mode, and another user may be unable to regain entry to the system. If they do not use the password regularly, they are likely to have forgotten it; even worse: a malicious user

might have changed the password without telling anyone, which will cause a much longer delay in regaining access to the system.

This kind of scenario generation – going from an identified difficulty to generate situations in which it might occur – is relatively novel, and provides an explicit bridge between usability evaluation and domain analysis. The usability evaluation identifies potential difficulties; the scenarios provide a focus for assessing the likely impact of those difficulties in actual use. This use of scenarios contrasts with that of Rosson and Carroll (2002), or with the 'Use Cases' of UML (Fowler, 1997): in those cases, scenarios are used to represent domain tasks and situations, to guide design or evaluation; in our case, scenarios are being generated from evaluation to pose questions about the domain.

User modelling: probing deeper

One important point that emerged from this initial analysis was that separating the facility for enabling / disabling lockout from that for activating it was likely to make the system less predictable to users. A related question was how operator and service level access relate to each other: it was possible that the existence of two different levels of security, to access different functionality, would also cause difficulties. The design problem was not yet stated explicitly enough to support detailed analysis; to understand these issues better, a formal model of the system was constructed.

Both a specification of the device (in natural language) and a corresponding PUM model, describing the knowledge the user needs to work with the device, are presented in Appendix 1. This "conceptual model" of the device encapsulates the knowledge of entities, relationships and operations that a user would have to know about to be able to use the device effectively. As these models show, although the lockout feature can be described quite clearly and concisely from a device perspective, it is much more difficult to create a plausible user-centred model.

The formal modelling enabled the analyst to identify potential usability difficulties with the new design:

1. It is not possible for the service-level user (who has the highest level of authorisation) to predict whether pressing lockout will simply disable service level access or whether it will activate lockout.
2. The user cannot immediately tell what the current access level is set to.
3. The user has to be aware of, and able to manipulate, more concepts than is strictly necessary. In particular, the three concepts of access-level, lockout-activated and lockout-enabled could easily be combined into one.

The first of these features is a more general statement of the problem illustrated by negative scenario (c) above; the second feature is a more general statement of the problem illustrated by (a); the third is new: the process of constructing a model forced the analyst to be explicit about features of the design that were unclear from earlier informal analysis.

Constructing the user-centred PUM model forced the analyst to describe the representation embodied within the device in detail, in a way that helped to identify possible simplifications. While observation would identify the surface behaviours, it would not give insights into the underlying causes for those behaviours. However, this stage of analysis was the most expensive (in terms of analyst time), and in many situations the additional insights obtained would not be worth the additional costs (see table 1 below for an estimate of time taken for each stage).

Discussion

Analysing the problem rigorously from a user perspective highlighted some potential difficulties, particularly concerning the underlying conceptual structure of the device. In addition, the usability evaluation raised some issues that could only be resolved by reference to the way the system is commonly used.

The predictions produced by the usability analysis were conditional; they stated that under certain conditions, particular issues could potentially be a problem. However, as a discretionary feature that clearly posed usability difficulties, our expectation was that the security feature would not be used. It was necessary to investigate the actual context of use to validate this prediction and to see to what extent these issues were in fact problems.

Stage 3: domain investigation

From the usability analysis, various questions about the context of use could be identified. In this case, the questions posed included:

- How is lockout used (i.e. what is policy on when it is and is not activated, and who uses the system functionality)?
- How well does practice match policy?
- Have there been critical incidents relating to lockout, and what were their consequences?
- What is the range of expertise of the users?

These questions related to how the device was used in context; this made a site visit the most appropriate environment for knowledge elicitation. While a site visit to just one customer would not give valid general answers, it would give indicative answers for a "test of concept" study.

Site visit

We present a brief description of the site visit to illustrate the new kinds of information that emerged through the visit, and how questions raised in the earlier analysis could be answered, at least in part. Data collection was based on observation and informal interviewing, including questioning about critical incidents.

The site visited was a large factory which packaged a variety of dried foods. The site used 12 printers, all of the "old" style, but the management were anticipating upgrading some when the new product became available.

The workforce was organised into four teams, each allocated to one area of the site. Each team consisted of packers, operators and team leaders, under the overall control of the manager. Most of the printers were operated by two of the teams, so the leaders of these teams had much more highly developed expertise in the operation and maintenance of the printers than the leaders of the other teams. Operators did routine tasks such as retrieving a message from the message store, modifying it, and starting and stopping the machines. Packers were never expected to touch the printer equipment. Setting up and fault finding were generally done by team leaders.

Referring back to the initial interviews, the site visit provided substantially more detail: people are not classified simply as 'users' or 'non-users', but the organisational structure is such that there is generally appropriate expertise available while also limiting the number of people who can legitimately perform particular tasks with the device. One consequence of this is that the predicted possible problems with users forgetting how to work the device turned out not to be a problem.

On entering the factory, one immediate example of taken for granted knowledge, involving the operating environment, became obvious: because the site was dealing with dry foods, the air contained a lot of fine powder, which made wearing spectacles impractical. This meant that staff members who needed reading glasses and who wanted to check the device's display – e.g. for monitoring codes – would have to stop what they were doing, take out their spectacles from under their protective clothing, put on the spectacles, read the display, and then put them away again under the protective clothing. Glancing at the display in passing to check the state of the device was not feasible for them. The device was also situated at about hip height, so users had to bend down to use it. While the need to have a WYSIWYG display had been mentioned in the interviews, the constraints of many operating environments (such as this one) had not; neither had the fact that there are two models of the current design: a high version that is of fixed location (which was the model demonstrated), and a low version on castors that can be moved easily but is inconveniently low for many users.

Security and password issues

Part of the answer to the question about security was immediately apparent on entering the factory. Almost all of the devices were situated next to teams of packers, making it extremely difficult for anyone to gain access to the device unnoticed. This was a passive work-around, but one of which the site manager was explicitly aware: when showing the investigators a device that was in an isolated part of the site, he described strategies which he had adopted to ensure security for that device.

As anticipated, the manager said that security was a potential problem – there had been one incident where a printed code changed mid-shift for reasons which were never satisfactorily explained, and another where a service engineering had forgotten to disable

the service password and someone else had changed it, so that no-one could access the system. This is very similar to negative scenario (c) above. He reported that there were stories from elsewhere in the food industry of considerable financial loss caused by malicious mislabelling; this corroborated the account given by Domino interviewees earlier.

The predicted possible problems with password lock-out were confirmed by the manager as being real problems. However, the problem appeared in two guises: firstly, it was easy to forget to re-set the device to the correct state after service level access, and this was identified as a risk for security breach; secondly, as activity in the factory changes, equipment is moved around from one packing line to another, so if passwords were used then it would be necessary for a large number of people to know them. This was given as a reason for not using operator-level password protection. While the first of these problems had been anticipated through the earlier analysis, the second had not.

Overall, the site visit helped confirm some predictions, identify some context-related issues missed from the analysis, and eliminate 'false positives' from the earlier analysis. This, in turn, helped in ascertaining the importance of usability issues. For example, the visit showed that whether or not security is a problem (for which usable security mechanisms are needed) depends on the location of a device within the broader context of use.

Debriefing with sales representative

Following the site visit, a short debriefing session was conducted with the Domino sales representative to obtain his view of how typical this particular customer site was. He confirmed that, in his experience, very few customers (perhaps as few as 2%) use the operator level password system. In his view, although many customers consider it as a factor in their purchasing decision, most later stop using it or decide not to use it after all. This additional interview helped check the generality of observations made during the site visit. If the aim had been to construct an accurate domain model of industrial printing, this would not have been an adequate substitute for further site visits, but for the purpose of this study it was an efficient substitute.

Stage 4: from problem to possible solutions

Identifying and understanding problems is useful, but it is also necessary to identify possible solutions. Proposing design solutions to particular problems involves understanding the problems then making a creative leap, often drawing on prior experience and using analogous evidence from other domains.

Examples of design suggestions

The problems and possible solutions generated in the course of this evaluation were reported to the design team, who were then asked for feedback on the findings. To

illustrate the kind of input that is possible into design, we present two from a longer list of design suggestions presented back to the design team.

As discussed, for devices that require user authentication, post-completion errors are a common problem, and active and passive workarounds are pervasive. One simple solution is derived from an analogous situation in another domain: cash registers in supermarkets have a normal level of access and a higher level of access, both controlled by locks and keys. The main problems in the printer domain were visibility of the state of the device and the related risk of leaving the device unsecured through forgetting to exit privileged access mode. These could both be overcome by using the supermarket approach of a physical key attached to the user's belt, so that if the user attempted to walk away while the key was still in the lock, they would not be able to do so.

A simpler and cheaper solution for service level users would be to disable service level access as soon as the user leaves the 'service' area.

Feedback from design team

A short report (3 pages) summarising our findings and design suggestions was delivered to the design team, followed by a meeting. Minor changes to the new system were made in direct response to the report; in particular, the security mechanism was modified so that, on exiting from the service area, the service password was automatically disabled. More major changes, such as revising the conceptual structure, are being considered for the next version of the product. The particular suggestion of using physical keys was rejected as being too expensive to implement in hostile (dusty or damp) environments, although a member of the design team reported using such a solution in a medical (i.e. safety-critical) application.

Perhaps most tellingly, the design team manager was keen to pursue the work further for the design of the next generation product, incorporating this style of work from the earliest stages of design. He could envisage a design process where software engineers and usability specialists worked closely together, each working to their own strengths, However, he could not envisage one in which any team member possessed the full skill set. He was not interested in the prospect of any of his team members being trained in user-oriented techniques, preferring to buy those skills in on an 'as needed' basis.

General Discussion

We now re-visit the original aims of this study, which was to incorporate domain information with an analytical evaluation technique that has traditionally viewed the user and device as a 'closed system'.

To uncover and explore issues in a controlled way, techniques were selected to expose the knowledge of different participants in the product lifecycle systematically. The selection of particular techniques to gather certain types of information, and maximise the validity of the data and analysis has been widely recognised (e.g. McGrath, 1984). The need for triangulation in design is discussed by Mackay & Fayard (1997). The use of techniques in

a particular order so as to maximise the explicit representation of design knowledge, as proposed and illustrated here, goes one step further: not only do the data sources triangulate; they also make as much as possible explicit. The main outcome of each stage of the process is summarised in table 1. This table also includes an indication of which author led each phase of activity and an informal estimate of the time taken. [Detailed diaries of time spent were not kept, so these figures are indicative, rather than definitive.]

Table 1 about here

As indicated in Table 1, the most time consuming phase was the detailed PUM analysis; the main deliverable from this phase was a more explicit representation of the underlying conceptual structure, which is of value, but in many cases would not be of sufficient value to merit the time invested. The PUMA Footprints yielded more insight for lower cost – certainly when applied by an expert. The insights regarding observability and predictability could have been gained using other techniques such as Cognitive Walkthrough (Wharton *et al,* 1994) or an appropriately designed empirical study; the recognition of the risk of post-completion errors would emerge less automatically from these techniques, but would be recognised by a suitably skilled human factors practitioner.

The device being studied here defines its own domain (rather than being introduced as an alternative way of achieving some pre-existing task). Therefore, a preliminary understanding of the domain could be gained by acquiring an understanding of the device. This understanding could not have been achieved without the company interviews, demonstrations and user-oriented analysis. In addition, this approach made as much as possible explicit, minimising the taken for granted knowledge of the investigators. Thus, *in this particular case*, we believe it was appropriate to conduct knowledge elicitation with company representatives and analytical evaluation prior to the site visit, as achieving a depth of understanding of the device and the usability issues was essential to efficient and effective use of the site visit. The analysis work helped focus the site investigation on pertinent issues. In other cases – for example, where the device is a well understood product and the issue was about its suitability for a particular domain of application – it would be important to include site visits earlier in the process.

The site visit contributed to an understanding of typical contexts of use and working practices; this enabled the investigators to validate points raised in the earlier interviews and clarify the situation where the two interviewees had different views. It would not, however, have been possible to directly identify the more detailed usability difficulties highlighted in the analytical evaluation – such as those caused by separating the facility for enabling / disabling lockout from that of activating it – without conducting a detailed empirical study within the context of visits.

One important mechanism developed for focusing the questions during the site visit was the generation of negative scenarios that identified conditions under which potential problems might become real problems, in order to assess their seriousness. In this

particular case, negative scenarios were generated by starting from the identification of information that was not observable, and generating situations in which the non-observable aspects of the state were not as expected; in principle, numerous site visits could be used to assess the likelihood of all scenarios, and hence the severity of the usability difficulties; in practice, with only one site visit, only the real risk posed by one of the scenarios (c) could be established. With different analytical techniques, negative scenarios could be generated from other identified difficulties, by considering the circumstances in which a possible difficulty could become an actual problem and then using site visits (or equivalent) to assess the likelihood of those circumstances arising.

Using the ACRE framework to guide the selection of knowledge acquisition techniques also enabled us to informally test it. The results obtained using the framework were as hoped; for instance, the designer and the trainer interviewed both appeared during the interview to have clear and complete understanding of how the current version of the device worked. However, when they demonstrated this version to the investigators, several features of the device appeared which had not been mentioned in the interviews, apparently as a result of failure of recall.

The site visit uncovered a variety of domain requirements which had gone unremarked previously. It was also possible, via the site visit, to check aspects of organisational and end-user behaviour in the context of use. For example, the physical position of the devices, near teams of packers made it difficult for anyone to misuse the devices unobserved; this made it unlikely that there was any significant difference between the manager's statement that there had been no significant problems with malicious misuse of the device (the 'front' version), and the behind-the-scenes reality (the 'back' version). In a site with a different layout, a claim of this sort might have required further checking, using other elicitation techniques such as indirect or participant observation.

Some predictions and techniques from the framework did not come into play. For example, the investigators were prepared for distortions in users' recall of how often particular problems occurred, and for significant differences between front and back versions at the site visited; in this case study, neither of these turned out to be an issue.

Conclusion

Assessing the usability of a device in context depends on an understanding of both human-computer interaction and the domain and context of use. Many design methods advocate an iterative design process that incorporates stages of requirements acquisition and usability evaluation (see, for example, Avison and Fitzgerald, 1995, for an overview of methods); however, within the process those stages are dealt with independently.

Recent approaches such as Contextual Design (Beyer & Holtzblatt, 1998) and Scenario Based Design (Rosson & Carroll, 2002) have contributed substantially to the development of techniques that start to bridge this gulf. The approach taken here has been different: rather than starting from the position that design culture and practice need to be completely overhauled, we have worked alongside software engineers who are skilled in

their own discipline, using established knowledge acquisition and evaluation techniques to introduce user and domain considerations to the established design process. In addition, in a design environment where new design evolves from existing products, we have incorporated a detailed usability analysis of the product with the analysis of context of use.

The approach reported here is not 'one size fits all': the order of techniques chosen here was appropriate for this purpose, but would not be for all. We have presented the rationale for the selection of techniques so that their appropriateness for other applications can be assessed. It is our intention that others, with their own repertoires of analysis techniques, can adapt the method applied here to their own purposes.

Although this study was limited by practical constraints (that the commercial design process with which we engaged was too far advanced to fully incorporate the work as it proceeded), it represents one contribution to the important but under-researched topic of how user- and domain-oriented evaluation can be effectively incorporated with each other and with commercial design practice.

Acknowledgements

We are grateful to Alan Jones, Jane Barlow and Peter Morris of Domino, and staff at Jacob Kraft Suchard for their co-operation in this study, and to Harold Thimbleby, Saul Greenberg, Kent Norman and anonymous referees for constructive criticism of earlier versions of this paper. This work was partially supported by EPSRC grant GR/L00391.

Bibliography

Adams, A. & Sasse, A. (1999) Users are not the Enemy Why users compromise security mechanisms and how to take remedial measures. *Communications of the ACM*, 42 (12) pp. 40-46.

Avison, D. E. & Fitzgerald, G. (1995) *Information systems development: methodologies, tools and techniques*. McGraw-Hill. Second edition.

Bellotti, V. (1989) 'Implications of Current Design Practice for the Use of HCI Techniques' in D. Jones & R. Winder (Eds.) *People and Computers IV, Proceedings of HCI'89,* 13-34. Cambridge University Press

Bellotti, V., Blandford, A., Duke, D., MacLean, A., May, J. & Nigay, L. (1996) Controlling accessibility in computer mediated communications: a systematic analysis of the design space. *HCI Journal*. 11.4 pp.357-432.

Blandford, A. E., Buckingham Shum, S. and Young, R. M. (1998) Training software engineers in a novel usability evaluation technique. *International Journal of Human-Computer Studies*, 45(3), 245-279.

Blandford, A. E., Butterworth, R. & Curzon, P. (2001) PUMA Footprints: linking theory and craft skill in usability evaluation. In M. Hirose (Ed.) Proc. Interact 2001. 577-584. Amsterdam: IOS Press.

Blandford, A. E. & Young, R. M. (1996) Specifying user knowledge for the design of interactive systems. *Software Engineering Journal*. **11.6**, 323-333.

Beyer, H. & Holtzblatt, K. (1998) *Contextual Design*. Morgan Kaufmann.

Boose, J.H., Shema, D.B. & Bradshaw, J.M. (1989) Recent progress in AQUINAS: a knowlede acquisition workbench. *Knowledge Acquisition* 1, 185-214.

Buckingham Shum, S. & Hammond, N (1994). Transferring HCI Modelling & Design Techniques to Practitioners: A Framework & Empirical Work. *Proceedings of HCI'94: People and Computers IX*. Cambridge University Press: Cambridge, pp. 21-36.

Burton, A.M., Shadbolt, N.R., Rugg, G. & Hedgecock, A.P. (1990) The Efficacy of Knowledge Elicitation Techniques: A comparison across Domains and Levels of Expertise, *Knowledge Acquisition* 2, 167-178,.

Byrne, M. D. & Bovair, S. (1997) A working memory model of a common procedural error. *Cognitive Science*. 21.1, 31-61.

Card, S. K., Moran, T. P. and Newell, A. (1983). *The Psychology of Human Computer Interaction,* Hillsdale : Lawrence Erlbaum.

Carroll, J. M. and Rosson, M. B. (1992) Getting around the task-artifact cycle: how to make claims and design by scenario. *ACM Transactions on Information Systems*, 10(2), 181-21.

Coble, J. M., Karat, J. & Kahn, M. G. (1997) Maintaining a Focus on User Requirements Throughout the Development of Clinical Workstation Software. In *Proc. CHI'97*. 170 – 177.

Connell, I. W. & Hammond, N. V. (1999) Comparing Usability Evaluation Principles with Heuristics: Problem Instances vs. Problem Types. In M. A. Sasse & C. Johnson (Eds.) *Human-Computer Interaction INTERACT'99*. 621-629. Amsterdam: IOS Press.

Denzin, N.K. and Lincoln, Y.S. (eds.) (1994) *Handbook of Qualitative Research.* Sage, London.

Dix, A.J., Finlay, J., Abowd, G. & Beale, R. (1998). *Human-Computer Interaction,* Hemel Hempstead: Prentice Hall International.

Domino (2002) Information page on A300 printer (viewed on 23rd May 2002): http://www.domino-printing.com/industry/products/a_series/A300/

Flanagan, J.C. (1954) The Critical Incident Technique *Psychological Bulletin*, 51, 327-358

Fowler, M. (1997) *UML Distilled.* Addison-Wesley, Reading MA.

Goffman, E. (1959) *The Presentation of Self in Everyday Life*, New York, Doubleday.

Good, J. P. & Blandford, A. E. (1999) Incorporating Human Factors Concerns into the Design and Safety Engineering of Complex Control Systems. In J. Noyes & M. Bransby (Eds.) *People in Control: An International Conference on Human Interfaces in Control rooms, Cockpits and Command Centres*, IEE Conference Publication Number 463, IEE, London. Pages 51 - 56.

Gough P.A., Fodemski F. T., Higgins S. A. & Ray S.J. (1995) 'Scenario - an industrial Case Study and Hypermedia Enhancements', *Second IEEE International Symposium On Requirements Engineering*, IEEE Computer Society Press.

Gray, W., John, B & Atwood, M. (1993) 'Project Ernestine: Validating a GOMS Analysis for Predicting and Explaining Real-World Task Performance', *Human-Computer Interaction*, 8. pp 237-309.

Grice, H.P. (1975) Logic and Conversation. In Cole, P. & Morgan, J.L. (eds.) *Syntax and Semantics 3* New York, Academic Press

Hollnagel, E. (1993). The phenotype of erroneous actions. *International Journal of Man Machine Studies*.39. pp. 1-32.

Hollnagel, E. (1998). *Cognitive Reliability and Error Analysis Method (CREAM)*. Oxford : Elsevier.

John, B. & Kieras, D. E. (1996). Using GOMS for user interface design and evaluation: which technique? *ACM Transactions on CHI*.1-30

Kelly, G.A. (1955) *The Psychology of Personal Constructs*, New York : W.W. Norton,.

Mackay, W. E. & Fayard, A-L. (1997) HCI, Natural Science and Design: A Framework for Triangulation Across Disciplines. *Proc. ACM DIS'97*. pp. 223-234.

Maiden, N.A.M. & Rugg, G. (1996) ACRE: a framework for acquisition of requirements *Software Engineering Journal*, pp. 183-192,

Mayhew, D. (1999) *The Usability Engineering Lifecycle: A Practitioner's Handbook for User Interface Design*. San Francisco: Morgan Kaufmann.

McGrath, J. (1984) Methods for the Study of Groups. In J. McGrath: *Groups: Interaction and Performance*. Prentice Hall, NJ. pp 28 – 37. Reprinted in R. Baecker (Ed.) (1993) Readings in Groupware and Computer-Supported Cooperative Work. Morgan Kaufmann. pp. 200 – 204.

Newman, W. M. & Lamming, M. G. (1995) *Interactive System Design*. Harlow : Addison-Wesley.

Nielsen, J. (1994) Heuristic Evaluation. In J. Nielsen & R. Mack (Eds.), *Usability Inspection Methods* pp. 25-62. New York: John Wiley.

Nielsen, J. & Mack, R. (1994) (Eds.), *Usability Inspection Methods* New York: John Wiley.

O'Neill, E. (2000) *User-Developer Co-operation in Software Development*. Springer-Verlag: London.

Robertson, S. (2001) Requirements trawling: techniques for discovering requirements. *International Journal of Human–Computer Studies*. 55.4. 405-421.

Rosson, M.. B. & Carroll, J. M. (2002) *Usability Engineering*. San Francisco: Morgan Kaufmann.

Seger, C.A. (1994) Implicit learning *Psychological Bulletin*, **115** (2): 163-196.

Shadbolt, N.R., & Wielinga, B.J. (1990). Knowledge based knowledge acquisition: the next generation of support tools. In B.J. Wielinga, J. Boose, B. Gaines, G. Schreiber, & M.W. van Someren, (eds.) *Current Trends in Knowledge Acquisition*. IOS Press, Amsterdam, pp. 313-338.

Sommerville I., Rodden T., Sawyer P., Bentley R. & Twidale M. (1993) Integrating Ethnography into the Requirements Engineering Process *Proceedings of IEEE Symposium on Requirements Engineering*, IEEE Computer Society Press, 165-173.

Spencer, R. (2000) The Streamlined Cognitive Walkthrough Method, Working Around Social Constraints Encountered in a Software Development Company, *Proc. CHI'2000*. 353 – 359.

Tulving, E. (1983) *Elements of episodic memory*. Oxford University Press, Oxford.

Wharton, C., Rieman, J., Lewis, C. & Polson, P. (1994). The Cognitive Walkthrough Method: A Practitioner's Guide. In J. Nielsen and R. Mack, Ed. *Usability Inspection Methods*, 105-140. Wiley: New York.

Young, R. M., Green, T. R. G., & Simon, T. (1989) Programmable user models for predictive evaluation of interface designs. In *Proceedings of CHI '89*. New York: ACM.

Appendix 1: Formal usability evaluation

To clarify the design, a device specification of the effects of user actions was generated in natural language:

1. If user has pressed lockout and lockout is enabled then disable all functions until user enters a correct password.
2. Else if user has pressed lockout and lockout is disabled then permit access to operator level functions only.
3. Else if last password entered is operator level then permit access to operator level functions only.
4. Else if last password entered is service level then permit access to all functions.

To assess this from a user perspective, a "conceptual model" was constructed to encapsulate the entities and relationships that a user would have to know about to be able to use the device effectively. We consider three tasks: performing some unspecified operation ("X") on the device (for example, checking ink levels or modifying the label code), activating lockout and disabling service level access.

A user's conceptual model of this system would have to include the following essential concepts:

```
setting:                on / off
user-access-mode:       none / operator / service
password:               operator / service
device-function:        store-label, retrieve-label, check-ink-level,
                        edit-label, etc.
```

The system state is described in terms of predicates referring to these concepts:

```
access-level-set(user-access-mode)          ; current access level
password-known(password)                    ; user knows certain password(s)
permits-access(password,user-access-mode)   ; password activates a certain access level
required-access-level-is(device-function,user-access-mode)
                                  ; device-function can only be invoked at certain access level
lockout-enabled(setting)                    ; is lockout enabled?
lockout-activated(setting)                  ; is lockout activated?
```

The representation chosen here is intentionally user-centred. There are constraints that users should also be aware of. For example, lockout cannot be activated unless it is enabled; leaving service mode activates lockout if lockout is enabled; etc. These constraints are expressed through the modelled user's knowledge of operations.

We start by specifying that in order to achieve some device-function (e.g. editing a label), the user has to know what access level is needed and has to have set that level (or a higher level of access).

```
Operation perform-function (device-function: X)
       User-purpose:    done(X)
       Filter:          required-access-level-is(X, L)
       Precondition:    access-level-set(M) & M ≥ L
       Action:          do X
```

The precondition to the `perform-function` operation is that the user knows that the access-level is high enough. Assuming that the current access-level is known to be lower than that needed (though there may be situations in which it is not known, and the user has to set about discovering it), the user needs to know the password that corresponds to the required level of access:

```
Operation set-access-level(level: L)
       User-purpose:    access-level-set(L)
       Filter:          password-known(P)
                        permits-access(P, L)
                        access-level-set(M) & M<L
       Action:          enter-password(P)
```

We consider also the two tasks of activating lockout and disabling service level access. There is some difficulty in defining the purpose of activating lockout. The important point is that it only works if lockout is enabled. Considering the definition of disabling service access, we note that there is the possible side-effect of activating (total) lockout if lockout is enabled. Thus the service-level user who wants to follow service-level activities with operator-level ones might reasonably remain in service level mode until all activity has been completed:

```
Operation activate-lockout
       User-purpose:    lockout-activated(on)
                        OR access-level-set(none)
       Precondition:    lockout-enabled(on)
       Action:          press "lockout" button

Operation disable-service-level-access
       User-purpose:    access-level-set(M) & M≠service
       Action:          press "lockout" button
```

This more detailed description has exposed a variety of user difficulties with the current representation. Ideally, the system would present, and allow the user to manipulate, these aspects of the state in a consistent way. Alternatively, the underlying model could be simplified – for example, by conflating the three concepts of lockout, lockout-state and user access mode into one concept of "current access level", for which the possible values might be: all access barred; access to operator areas only; access unrestricted. Then the issue becomes one of how to change the current access level, with a secondary issue of whether the default state at a particular site is to have all access barred or to permit access to operator areas only.

	Stage	Purpose	Issue	Lead analyst	Estimated time
0	Preliminary meeting	Outline of designs (current & new) Issue identification	Post-completion problem identified	GR / AB	1-2 hours
1	Interviews (designer & trainer)	Data gathering. Probing issues identified	Critical incident recounted. Designer believes password protection is a solution; trainer believes procedures within organisation are needed.	GR	2 * 30 minutes interviews + 4 hours analysis
	Demonstration of current system	Data gathering. Deeper understanding of design.	Design of current system clarified. Problem of 'escaping' too high experienced	AB / GR	30 mins + 3 hours analysis
	Presentation of new system	Deeper understanding of proposed design	Re-design believed by designer to fix problem	AB / GR	1 hour
2	Usability evaluation based on PUMA footprints	Quick identification of usability difficulties	Observability and predictability identified as problems	AB	1 hour
	Negative evaluation scenarios	Identify situations in which usability difficulties would manifest themselves	Potentially difficult situations identified – to be investigated later	AB	1 hour
	PUM user modelling	Deeper analysis of conceptual model	Redundant concepts needed by users	AB	7 hours
3	Site visit: observation and informal interview with user	Comparing current understanding to user experience in organisational setting	Security mechanism is not used; workarounds are almost satisfactory.	GR / AB	2 hours + travel time
	Debriefing with sales representative	Assessing generalisability of findings	Not used by '98% of customers', but helps marketing	GR / AB	30 minutes
4	Report back with design suggestions	Assess appropriateness of design suggestions, within the full design context	Other design constraints identified; some changes made.	AB / GR	2 days (report writing) + 1 hour meeting

Table 1: outline summary of stages, issues, roles and approximate time taken.