ETH *Eigenössische Techniche Hochschule Zürich*

IHA Institut für Hygiene und Arbeitsphysiologie IHA

Swiss Federal Institute of Technology
Institute for Hygiene and Applied Physiology
Man-Machine Communication Dept.
Zurich, Switzerland

Technical University of Gdańsk
Faculty of Management and Economics
Ergonomics Dept.
Gdańsk, Poland

Institutional Partnership Project 7 IP 051780
supported by *Swiss National Science Foundation*
Berne, Switzerland

TRANSFERRING USABILITY ENGINEERING TO INDUSTRY

International Workshop
Proceedings

Edited by:
Marcin Sikorski
Matthias Rauterberg

11-14 June, 1998
Technical University of Gdańsk, Poland

Technical University of Gdańsk
ul. Narutowicza 11/12, 80-952 Gdańsk, Poland

ISBN 83-902559-8-7

CONTENTS

PREFACE

This International Workshop "Transferring Usability Engineering to Industry" has been organised as one of the first events in Poland, dedicated to usability issues.

Recent economical changes in Central-Eastern Europe, together with rapid development of information technology, raised the question of usability issues for computer-supported work in Polish companies. Computer systems nowadays seem to be developed much faster and in more cost-effective way than before. However, the number of dissatisfied users still remains high: also company managers complain that systems are difficult to learn and to operate, and that they do not support business processes in a sufficient degree.

Mission-critical systems for banking, corporate or military computing, as well as open systems for public use, for education or entertainment, need to include the limitations and preferences of the users from the very early concept. Access to modern technology allows today to design systems in much easier way, but it is also much easier to design *a very bad system* in much shorter time.

Because design for usability needs a specialised knowledge and skills, companies designing interactive systems often employ external consultants to verify and improve the quality and usability of systems design. However, still there is a big number of systems of poor usability, or disliked by the users, because usability issues have been lost somewhere in the design process when taking key design or costing decisions.

This workshop is intended to gather usability professionals, experienced in research and consulting, in order to present their experiences, and to improve their skill of transferring usability solutions for industrial clients.

The second goal of this workshop is to compare different educational programmes on usability in order to develop the most optimal educational scheme for professionals from growing software industry in Poland. Presentation of various foreign experiences is also important for studying usability-relevant crosscultural issues – so important in today's global and networked economy.

Organisation of this workshop was enabled by an Institutional Partnership Grant No. 7 IP 051780 from the Swiss National Science Foundation whose support is hereby highly appreciated.

Workshop organizers:

Dr Marcin Sikorski
Technical University of Gdańsk
Faculty of Management and Economics
Ergonomics Dept.
Gdańsk, Poland

Dr Matthias Rauterberg
Swiss Federal Institute of Technology
Institute of Hygiene and Applied Physiology

Zurich, Switzerland

Section I

APPROACHES

Usability from the top and from the bottom

Jurek Kirakowski *jzk@ucc.ie*
Bozena Cierlik *bozena@ucc.ie*
Human Factors Research Group, UCC, Ireland

For some time now, it has become apparent that there exist at least two, if not three senses of the term 'usability testing'. It is the purpose of this paper to make some remarks about these different senses in which the term is used.

It has been understood for some time now that there exists a difference between usability testing carried out with reference to sets of rules or guidelines of desirable features of the interface, and usability testing carried out with reference to the actual experience of end users with the product being evaluated. This difference was noted as far back as the early 1990s, for instance, in the ESPRIT MUSiC project reference model [1] where it was observed that adherence to rules and guidelines and evaluation with reference to these rules and guidelines may prevent a disaster in terms of usability, but it would not necessarily help to create a really good, usable interface. The ISO 2941 standard [2] from parts 12 onwards embodies such an approach in which well-contextualised guidelines are given under the headings:

- screen appearance
- help
- HELP!!!

Some researchers have interpreted this to mean that the process of certifying that an interface demonstrates a minimum acceptable standard involves
(a) identifying which rules are applicable
(b) assessing to what extent the application demonstrates adherence to the applicable rules.
For instance the Lloyd's Register proposed 9241 accreditation standard parts of which have been included in the Telematics Application Programme project INUSE methods collection [3] leans fairly heavily on this procedure, although the procedure also states that in cases where there is extreme non-applicability, a 'performance-centered' approach as suggested by part 11 of the standard should be used.

However, even if it is acknowledged that severity of rule infraction should also be taken into account, such procedures are unsatisfactory as assessment procedures because they imply that the operation of adding (possibly weighted) scores of degree of conformance to a set of rules is a meaningful mathematical operation, leading to an interpretable result. A pilot of an aircraft may have satisfied 99% of the checklist items for a landing, for instance, but the landing has a high risk of disaster if he has failed to check the item 'ensure wheels are down'. Even if this item is weighted, for instance at a level of 20%, so that omission of this item yields a checklist score of 80% if all the other items are complete, the question of balancing this item against a host of less critical items (for instance such as 'ensure passengers have fastened safety belts') still remains.

Assessing software quality with reference to checklist items implies that checklist items form a homoscedastic (proceeding by equal numerical intervals) scale, whereas in reality this is usually far from the case. Checklists can be very useful for carrying out formative assessments (in the sense originally popularised by Scriven [4]) but they are dangerous instruments for summative assessment. The series of checklists developed by Ravden and Johnston [5] acknowledges this difficulty, and the authors recommend that the checklists should not be used in such a summative sense. The EVADIS procedures [6] present a highly structured approach to this kind of usability testing, but the mathematical underlay has been questioned (see for instance Kirakowski's paper on the use of questionnaire methods [7]).

The second sense in which 'usability testing' is used involves the experience of end users with the application. This experience may be reported either in a qualitative manner, that is, using verbal or iconographic descriptions, or it may be reported in a quantitative manner, relying on a mapping of attributes of user performance and judgement to the scale of natural numbers.

At its simplest, the qualitative version of this approach involves using concurrent verbalisations as suggested by Monk et al [8] or a collection of video clips which actually show instances of user difficulties with parts of the interface. This approach, especially the use of video sequences, has been found useful for communicating to designers problems with the application, so-called 'usability bugs.'

Quantitative approaches to usability testing with end users involve measuring aspects of user performance. These approaches are usually summarised under the headings:

- Efficiency
- Effectiveness
- Satisfaction

following the 9241 part 11 definition of the term 'usability.' Efficiency is defined in terms of measures such as time taken by the user for different kinds of operations; effectiveness is defined in terms of degree of conformance of the end result of the user's actions to the desired end state of the interaction, and satisfaction is defined in terms of the user's subjective experience of the application. There is another quantitative measure that is sometimes recommended, and that is the measure of Cognitive Workload [9]. The theory behind Cognitive Workload assessment is that while performance may reach a certain objectively defined level, the internal cost to the user is not assessed. Thus high efficiency may be demonstrated for a certain set of tasks, but if the cost to the user in terms of mental stress is high, then this efficiency cannot be sustained over long periods. Cognitive Workload may be subsumed under 'satisfaction' but it represents a dimension of experience with the application that is different enough from satisfaction to warrant its inclusion as a separate, fourth approach to the evaluation of performance-based usability assessment.

We shall return to quantitative approaches shortly, but it is important at this stage to mention another approach to usability testing which is part way between a checklist type approach and a performance approach. This is the concept of testing the application by assessing to what extent it conforms to a set of heuristics, or general principles which should underlie the development of a usable application.

The locus classicus of heuristic analysis is the work of Molich and Nielsen [10], although many variants of the set of heuristics are now in use, especially with regard to the assessment of usability of web sites. Many of these are summarised and collated in a recent paper by Bevan et al. [11]. In carrying out heuristic analysis, the evaluators agree among themselves as to how they will interpret the heuristics, and will then carry out an appraisal of the usability of the application following these heuristic principles. Heuristic evaluation is well known for its propensity of generating large lists of usability defects, and the best heuristic procedures involve teams of evaluators, including end user representatives, who before they conclude their evaluation, agree on an ordering of severity of defects found. An excellent example of this kind of procedure is the Cello method developed by Claridge and his co-workers at Nomos AB, which procedure is represented in the above-mentioned INUSE project methods collection. However, it is our observation that usually, heuristic evaluation is done rather informally and that empirical validation of heuristic principles is done relatively infrequently. A good empirical analysis of heuristic principles is presented by Scapin and his colleagues [12].

Returning to quantitative approaches. What do they tell the evaluator? In their plainest and least-developed form, quantitative approaches supply a 'figure of merit' which is usually averaged over a representative sample of users. A good example of such a figure of merit is the NPL metric, Relative User Efficiency, which states as a percentage the performance of a representative sample of users with respect to the performance of an expert user. Thus if RUE is given as 100% we can say that the users sampled equal the performance of the expert user.

Quantitative measures of satisfaction usually attempt to break down the overall concept of 'satisfaction' into a number of smaller sub-dimensions, although some fairly robust and simple uni-dimensional assessments of satisfaction have been developed. See for instance the SUS scale [13]. Sometimes these sub-dimensions are created a priori from expert judgement. The Software Usability Measurement Inventory [14] works with five sub-dimensions or sub-scales of the satisfaction concept, which have been empirically obtained by factor analysis of large numbers of statements that end users make about the usability of a computer application. These sub-scales are given as:

- Efficiency
- Affect
- Helpfulness
- Control
- Learnability

Thus the usual approach for satisfaction measures is to provide a satisfaction profile of the application. In the case of SUMI, the sub-scales have been developed so that the numerical values obtained can be interpreted with reference to other software products in the SUMI database. Values on SUMI sub-scales of less than 50, for instance, show that the application falls below the average expected value.

Cognitive workload measures, such as the SMEQ questionnaire [15], relate the user's perceived mental effort to a set of verbal anchors which describe the user's effort as being, for instance 'effortful', or 'tremendously effortful'.

To our knowledge, SUMI, and its descendants, MUMMS (Measurement of Usability of Multi Media Software) [16] and WAMMI (Web Analysis and MeasureMent

Inventory) [17] are the only quantitative measures of usability which allow a direct semantic interpretation of the data, although the SMEQ anchors are also useful indicators in terms of end users' experience.

One peculiarity of performance measures must be noted at this point: it is important that if the data being gathered is to be in any sense representative of what end users will experience when the application is used in the 'real world' outside the development environment, then the context in which the application is tested must be related to the context in which the application will eventually be used. In concrete terms, this implies that users who test the application must be representative of real-world users; the tasks they carry out in order to gain experience with the application must be representative of the tasks that the application is designed to support; and the social, technical, and organisational environment in which the testing is carried out must represent intended real world environments.

At the very least, the context of testing must be clearly stated so that an independent evaluator may come to their own conclusions about the extent to which the performance evaluation says something meaningful about the context of usage of the application in the real world.

It is now appropriate to bring all these approaches to usability testing together. We see that there is a useful dividing line that can be drawn through all the approaches mentioned so far. On one side of the line we have user testing approaches which focus on the details of the application, and are orientated towards the identification and fixing of 'usability defects' in the application. A major concern of practitioners of this kind of user testing is to involve the design and development teams in the consumption of the usability report. There is frequently resistance especially in organisations with a low level of maturity in usability engineering to this kind of input, and usability engineers working at this level often have a substantial selling job to do. Rubin [18] gives a good collection of advice from an expert at this kind of work.

On the other side of the line we have user testing which focuses on the overall usability quality of the application, without usually going into specifics as to what are the root causes of poor (or high!) quality. The consumers of this kind of report are usually managers and personnel involved in quality management, as well as purchasers and end user representative organisations. Oddly enough, reports of this kind are usually extremely well received by their audiences who sometimes show a less than critical appraisal of the weaknesses of usability metrics, that is, what usability metrics don't tell you. For instance, we have sometimes noticed that even a modest improvement over the expected average for SUMI scales is accepted with complacency by the client as evidence that the 'product is usable'; it is important in these circumstances to point out that there may still be a long way to go, and simply being somewhat better than the rest of the market is actually no hallmark of quality.

In fact, individual metrics such as SUMI can be used to make quite deep diagnoses of what is going wrong with an application, leading to bug fixing. However, their main characteristic is that the diagnoses are usually strongly ordered in terms of severity, and sometimes the most severe problems of an application are not found in the minutiae of the screens or interactions, but in rather broad design decisions that may take a lot more effort to fix.

What we have attempted to argue in this paper is that there are at least two levels at which the concept of user testing is applied. One is a pragmatic approach, near to the level of code generation for specific features of the application. Practitioners at this level rarely use metrics or formal testing, simply because the small problems can be uncovered relatively quickly, and once noted, stand out clearly. The important selling job to be done here is to persuade the development team that these problems are really problems that their users are going to face, and that these are not simply subjective whims of one or two evaluators.

The other is an approach similar to quality measurement and management. Practitioners at this level are usually careful to define the scope and procedures they will use in order to make real quality statements. The context of testing is important to practitioners at this level. The important selling job to be done here is to bring the client to the realisation that these metrics are indicators of problems with the application, and that if necessary, more investigative work needs to be done to focus the development effort on improving the quality of use of the product.

References

1. Kelly M., (ed.) (1994). *MUSiC Final Report Parts 1 and 2: the MUSiC Project.* Brameur Ltd., Hampshire, UK.
2. ISO 9241 (ed.) (1991). *ISO 9241 Ergonomic Requirements for Office Work with Visual Display Terminals.*
3. D6.2 Inuse Methods Handbook v1.1, ed. N.Bevan, NPL 1997.
4. Scriven M., Tyler R., Gagne R.(eds.) (1972). *Perspectives of curriculum evaluation.* Chicago Rand McNally&Co.
5. Ravden S.J., Johnson G.I., (1989). *Evaluating Usability of Human-Computer Interfaces: A Practical Method.* Ellis Horwood, Chichester.
6. Reiterer H., and Oppermann R., (1993). *Evaluation of user interfaces: EVADIS II -- a comprehensive evaluation approach.*
7. Behaviour and Information Technology, 12.3, 137-148. 7. See: http://www.ucc.ie/hfrg/questionnaires/sumi
8. Monk A., Wright P., Haber J., Davenport L., (1993). *Improving Your Human-Computer Interface. A Practical Technique.* New York, prentice Hall.
9. Houwing E.M., Wiethoff M., Arnold A., (1993). *Cognitive Workload Measurement. Instructions and Background Materials.* TUD Delft (WIT Lab).
10. Molich R., Nielsen J., (1990). Improving a Human-Computer Dialogue. *Comm. ACM* 33.3, p.338-344.
11. Bevan N., (1997). Usability Issues in Web Site Design. *Proceedings of HCI International'97*, San Francisco, 24-30 August 1997, Elsevier.
12. Bastien J.M.Ch., Scapin D.L, *Ergonomic criteria for the evaluation of Human-Computer Interfaces*, INRIA No156 (June 1993).
13. Brook J., (1988) *System Usability Scale.* DEC UIA A/D Group, HICOM Note 198.5.
14. Kirakowski J., Porteous M., Corbett M., (1995) *SUMI User Handbook.*
15 Houwing E.M., Wiethoff M., Arnold A.G., (1993). *Introduction to cognitive workload Measurement.* TU Delft.
16. See: http://www.ucc.ie/hfrg/questionnaires/mumms
17. See: http://www.ucc.ie/hfrg/questionnaires/wammi
18. Rubin J., (1994) . *Handbook of Usability Testing.* New York, John Willey&Sons Ltd.

Software Ergonomics in Practice: The Importance of Acceptance-Related Issues

Marc Hassenzahl & Jochen Prümper & Edmund Buchbinder
Büro für Arbeits- und Organisationspsychologie,
Berlin, Germany (*hassenzahl@bao.de*, *buchbinder@bao.de*)
Fachhochschule für Technik und Wirtschaft (FHTW),
Berlin, Germany (*pruemper@bao.de*)

1 Introduction

In the past two decades the field of software ergonomic design has made substantial progress; now providing rich resources of frameworks, tools and techniques waiting to be applied in commercial projects. However, from the perspective of the usability consultant such projects are often problem-centred, small budget consultations, requested as "last hope", instead of integrated, well-planned collaborations of system developers and the consultant.

What are the reasons for this? One answer may lie in the issues related to the *content* of the field. Theories, models and methods constituting software ergonomic design may not be appropriate for commercial application. We disagree with this view. In the last two decades, theoretical and methodological progress was made, culminating in a consensus about the cornerstones of design: early prototyping, many cycles of design iteration, involvement of real users and co-operation in interdisciplinary teams (Carroll 1997). The usability consultant can choose from a wide range of tools and techniques to select the methods that fit best the circumstances of the software ergonomic project on hand. Thus, *content-related issues* may not play the crucial role.

To understand why clients are often not willing to substantially invest in software ergonomics, we start with an example for problems we are confronted with in our daily work as usability consultants:

Imagine yourself being a usability consultant...: "A client asked you for help. A software system built for in-house use is not accepted by the users. You offered an evaluation of the system and promised to work out solutions for major usability problems revealed. When presenting your results and suggestions for improvements, your client is surprised and bluntly questions your work. He tells you that he believes the software system is already good enough as it is. You have to learn that nobody allows for additional time and money required for changes, although the need for these resources is obviously clear."

In analysing this scenario it becomes clear that *acceptance* of software ergonomic design goals and methods may play a crucial role. In our scenario the central goal, namely "improvement of the software system's quality by improving its usability" is not accepted. Otherwise the refusal of additional resources cannot be explained. The client obviously speculated that the mere presence of a usability expert will increase acceptance among users.

As the example shows, the behaviour of a client can be rather irrational. On the one hand he decides to hire a usability consultant in order to solve his problems, on the other hand sometimes even basic goals are not accepted, i.e. shared between the client and the consultant.

To understand these - in our view - *acceptance-related issues* is the subject of the remaining paper. First, we present a way of thinking about a client in order to understand the reasons for the problems described above (see section 2). Second, we will present examples for essential acceptance-related issues (see section 3).

2 A way of thinking about your client

A way of thinking about your client is as an organisation having a certain degree of maturity or being on a certain *evolutionary stage* referring to usability (Ehrlich & Rohn 1994; Nielsen 1994).

This view has some important implications: First, the client's behaviour will be different depending on the stage he or she is in, therefore problems will be different, too. Second, the consultant's behaviour must take the degree of the client's maturity into account. Third, the client has at least hypothetical chances to evolve.

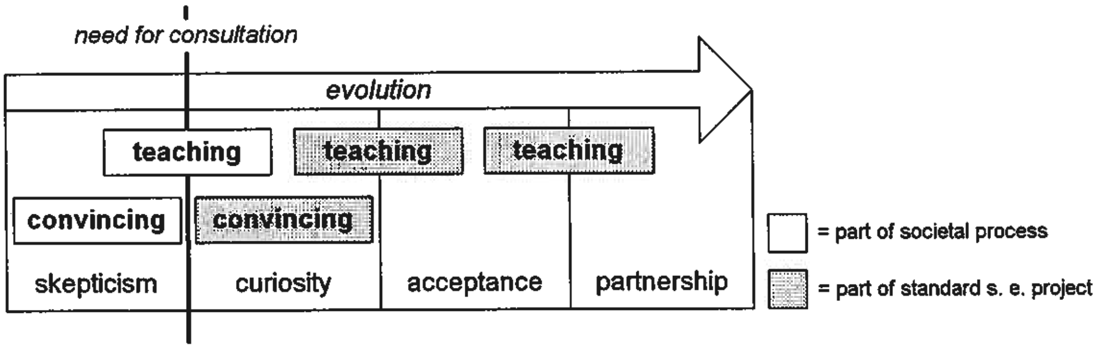


Figure 1: Evolutionary stages and acceptance-related activities

Figure 1 shows four stages of evolution (Ehrlich & Rohn 1994). In the "skepticism stage" software ergonomic consultation does not take place at all, either because usability is not important or the organisation is of the opinion that good usability can be achieved by the regular staff in the regular development process.

There may be two processes leading to the request of an usability consultation: First, the organisation has got an actual problem with a software system (e.g. acceptance is low) and somebody *convinced* it that the problem may be solved by usability consultation.

Second, the organisation does not have pressing problems concerning usability. However, it has *learned* about software ergonomics either from propaganda or formal education and finds it worthwhile to test whether the quality of its software systems may profit from the application of software ergonomic methods.

The two processes *convincing* and/or *teaching* (see Figure 1) are central to the development of a need for consultation, but they are societal and only under indirect control of the usability consultant. In other words, whether the "need for consultation"-barrier will be crossed is mainly a question of marketing and education.

By going beyond the "need for consultation"-barrier, an organisation enters the "stage of curiosity" and becomes a client. Projects at this stage are often characterised by small budgets. The client's request for consultation occurs late in the development process and is driven by special problems or breakdowns in the design process (i.e. problem-centred). Moreover, clients often do not know much about even basic goals and methods of software ergonomic design. Instead they sometimes possess a kind of "naive theory" of the way a usability expert is supposed to do his work, which often does not match reality well.

Thus, *convincing* remains an important activity at this stage - it even has to be *part* of the software ergonomic project. However, the type of convincing changes. Now that the general benefit of software ergonomic consultation is accepted by the client, the focus of the convincing process has to be put on the legitimation of the specific goals, methods and results.

Besides convincing, *teaching* must be an important part of a project at the "stage of curiosity" as well. From an organisational or "political" perspective projects at this stage are often problematic, but as regards professional contents not very demanding. This may be frustrating for the usability practitioner. Moreover, there is a potential risk that in future projects within the same organisation the practitioner has to cope with these problems over and over again. Thus, conditions for following projects have to be changed, i.e. the *organisation has to be changed*. Usability must become an element of organisational culture, and we consider the support of this change being an important part of the usability consultant's role (Hassenzahl, Prümper & Buchbinder 1998; Mayhew & Bias 1994). He is more than an expert for improving quality - he is a "change agent".

The "stage of curiosity" is followed by the "stage of acceptance". At this stage the organisation already had the opportunity to gain experiences with software ergonomic methods. Work at this stage is less problematic because of the general acceptance of goals and methods. The conditions may not be perfect, but the client is willing to do what is required to improve the usability of the software system. Hence convincing is not an important activity anymore.

However, *teaching* still remains important. The major objective of organisational change at this stage is to enlarge your client's independence. The application of software ergonomic methods must become a common practice, i.e. a part of organisational culture.

At the last stage, the "stage of partnership", usability is accepted and the organisation is able to act independently. There are different indicators of being in that stage. There may be a usability laboratory and/or dedicated groups of usability specialists. Within an organisation at this stage, software ergonomics is a part of early planning and the methods are used throughout the entire development process. The usability consultant is expected to act as a discussion partner when facing special problems.

3 Examples of convincing and teaching

As discussed above, *convincing* is an important activity especially when working for clients at the "stage of curiosity", whereas *teaching* is an important activity at all stages. In the following, we present examples of both activities.

3.1 Convincing

In the early stages (stages of "scepticism" and "curiosity") the organisation must be convinced that software ergonomics is necessary and useful. One of the most discussed strategies in this context is to reduce the costs of software ergonomic projects (Nielsen 1989), which is basically a "foot in the door"-strategy. The consultant speculates, that there will be a rising commitment to software ergonomics due to positive experiences.

Although the reduction of costs may be the crucial argument leading to the application of certain software ergonomic methods, to offer a cheaper method will not lead to any commitment by itself on behalf of the client. In other words, the client may have accepted the costs, but not the objectives and methods.

Sometimes it is necessary to convince a client that his software system is of poor ergonomic quality. In this case the client has to accept the consultant's basic goals, methods and standards. To accomplish this can be rather difficult:

In one of our projects, we carried out a user survey with the ISONORM 9241/10 questionnaire (Prümper 1993; Prümper 1997), which corresponds to our standard screening procedure. We used previously collected ISONORM-data to derive a coarse rule of thumb for determining whether the user's subjective assessment of the usability indicates acceptance or not. The data clearly showed the non-acceptance, i.e. discontent of the users.

However, this conclusion was questioned by the client. He argued that our standard might be too high and unattainable by an ordinary software system. Fortunately, through the years we collected ISONORM-assessments of many other software system. By that, we were able to demonstrate that other software systems which were used at the workplace (e.g. MS-Word 6.0) did meet our standards. The client devalued this *argument by pointing out that MS-Word was a standard system for word processing whereas their system was specifically developed for their purposes and basically a database. As a result, we compiled data which showed that even in these categories systems can be found that meet our standards.*

Even now, our client was still not convinced. He argued that the users (which are members of the client's organisation) are especially critical; indicating that the negative assessment may be a consequence of a general negativity of the users. We organised a second assessment, now asking users to judge a software system normally used at the workplace in the client organisation. On average their judgement of the usability was clearly more positive when judging other software systems than judging the software system in question. Moreover, the average judgement met our standards. Thus, the argument of a general negativity did not hold anymore. This was the moment for the client to give in and take steps to improve the usability of the system.

From a psychological perspective the client's behaviour in the example is easily explained: he may be reluctant to accept the usability problem because of the effort (time, money, emotions) already spent for a somewhat poor software system.

For the usability consultant it is important to be prepared for all types of convincing activities. Without proper arguments in the form of quantitative data (as used in our example) or a scientific "backing", some projects might end up quite unsuccessfully.

3.2 Teaching

Helping a client to evolve should be an important part of a standard software ergonomic project. If the usability consultant neglects those activities the client might remain at his actual stage. This is especially problematic at the "stage of curiosity". As pointed out above, work at this stage is often dominated by activities not directly related to the improvement of usability. Nevertheless, the consultant tries to improve the usability of the software as well. Given that the consultant is successful despite the bad circumstances, the client might learn from this that by and large the conditions he provided were adequate. The success works as a kind of reinforcement which lowers the motivation to provide better conditions for a following project.

A way to prevent this "getting stuck" at a certain stage is *teaching*. In our projects we made positive experiences with two types of activities. First, we constantly emphasise problems arising from inadequate conditions. Even a success will be contrasted with what could have been achieved under better circumstances. This is meant to demonstrate how an early integration of software ergonomic methods in the development would have eased the way to a high quality product. As a consequence, the client might be motivated to improve the way of dealing with usability questions, leading him or her to a higher stage of evolution.

Second, we always embed parts in our software ergonomic processes which are not primarily aiming at the improvement of a given system (e.g. theoretical views, background information). These educational activities are aimed at building up competence on the part of involved persons (e.g. project management, users, software developers). Competence is the basis for independence. Independence might improve conditions for software ergonomic projects by making the client capable of "asking appropriate questions at the appropriate time".

4 Conclusion

Software ergonomic projects can vary considerably with regard to problems encountered and budget. A way to understand the variation in behaviour of clients is to view them as being at a certain stage of evolution as proposed by Ehrlich and Rohn (1994) and Nielsen (1994). On the basis of the organisation's stage a usability consultant may predict a client's behaviour and the occurrence of stereotypical problems. This may help them to be prepared for acceptance-related activities, e.g. to take additional expenses caused by acceptance-related issues into account.

To our mind, there are different ways to further explore the presented approach. First, we may collect and classify "war stories" from commercial software ergonomic projects, building up more than just anecdotal knowledge about problems we might have to face. Second, we may find ways to reliably determine the stage a client is in, in order to anticipate problems linked to this stage. Third, we may further discuss methods for changing clients, i.e. changing their stage of evolution. Fourth, we may evaluate software ergonomic projects referring to the client's stage of evolution to understand more about the circumstances under which certain software ergonomic methods are or are not successfully applied.

5 References

Carroll, J.M. (1997). Human-computer interaction: psychology as a science of design. *International Journal of Human-Computer Studies*, **46**, p. 501-522.

Ehrlich, K. & Rohn, J. (1994). Cost-justification of usability engineering: A vendor's perspective. In R. G. Bias & D. Mayhew (eds.), *Cost-Justifying Usability*, p. 73-110. Boston, MA: Academic Press.

Hassenzahl, M., Prümper, J. & Buchbinder, E. (1998). Software-ergonomische Beratung in der Praxis - ein Beitrag zur Organisationsentwicklung. In A. Clermont & W. Schmeisser (eds.), *Sonderprobleme der betrieblichen Personal- und Sozialpolitik*. München: Vahlen (in press).

Mayhew, D. & Bias, R.G. (1994). Organizational Inhibitors and Facilitators. In R. G. Bias & D. Mayhew (eds.), *Cost-Justifying Usability*, p. 287-318. Boston, MA: Academic Press.

Nielsen, J. (1989). Usability engineering at a discount. In G. Salvendy & M. J. Smith (eds.), *Designing and Using Human-Computer Interfaces and Knowledge Based Systems*, p. 394-401. Amsterdam: Elsevier Science Publishers.

Nielsen, J. (1994). Guerrilla HCI: Using Discount Usability Engineering to Penetrate the Intimidation Barrier. In R. G. Bias & D. Mayhew (eds.), *Cost-Justifying Usability*. Boston, MA: Academic Press.

Prümper, J. (1993). Software-Evaluation based upon ISO 9241 Part 10. In T. Grechenig & M. Tscheligi (eds.), *Human-Computer Interaction*, p. 255-265. Berlin: Springer.

Prümper, J. (1997). Der Benutzungsfragebogen ISONORM 9241/10: Ergebnisse zur Reliabilität und Validität. In R. Liskowsky, B. M. Velichkovsky & W. Wünschmann (eds.), *Software-Ergonomie '97. Usability Engineering: Integration von Mensch-Computer-Interaktion und Software-Entwicklung*, p. 253-262. Stuttgart: B.G. Teubner.

Technology Transfer Methods Used by NPL as a European Usability Support Centre

Nigel Bevan *nigel.bevan@npl.co.uk*

National Physical Laboratory, Teddington, Middlesex, UK

NPL Usability Services has developed a range of methods, tools and training courses to assist industry in integrating methods for usability and user centred design into their development processes.

MUSiC Performance Measurement Method

NPL developed the Performance Measurement Method for usability evaluation [13] as part of the MUSiC project [4]. A detailed analysis of the context of use is used as a basis for planning a representative context of evaluation and realistic evaluation scenarios. Used in conjunction with SUMI [12], it provides design feedback combined with measures of effectiveness, efficiency and satisfaction.

The Performance Measurement Method has helped NPL work with organisations to clarify usability requirements and subsequently support evaluation of usability at appropriate points during design. This work has helped establish and train usability teams in a number of large organisations, applying the method in the development and testing of systems, and helping integrate the specification and evaluation of usability into quality management systems.

One major achievement of the Performance Measurement Method has been to proceduralise a process which has traditionally been the exclusive province of trained and experienced usability specialists. By documenting the activities in the evaluation process in detail, it can be taught to individuals who have no previous experience of human factors and psychology. At NPL, we have typically found that the training and technology transfer can be completed with 20 - 25 days of our effort. This includes three days training followed by assistance in the conduct of evaluations, tapering off as acceptable levels of individual competence for the various stages of the evaluation process are achieved. At the end of this process, an organisation has a skilled team of individuals who have demonstrated their competence in designing, carrying out and documenting user-based evaluations.

European Usability Support Centres

The INUSE project has set up a network of Usability Support Centres for the European information engineering industry, in conjunction with the RESPECT and MEGATAQ projects. A co-ordinated service incorporating common core methods is being provided by centres including NPL, HUSAT (Loughborough), Lloyds Register (London), University College Cork, Fraunhofer

IAO (Stuttgart), SINTEF (Oslo), Nomos Management (Stockholm), WIT Lab (Delft), CURE (Vienna) and other usability consultancies in Europe (see Figure 1).

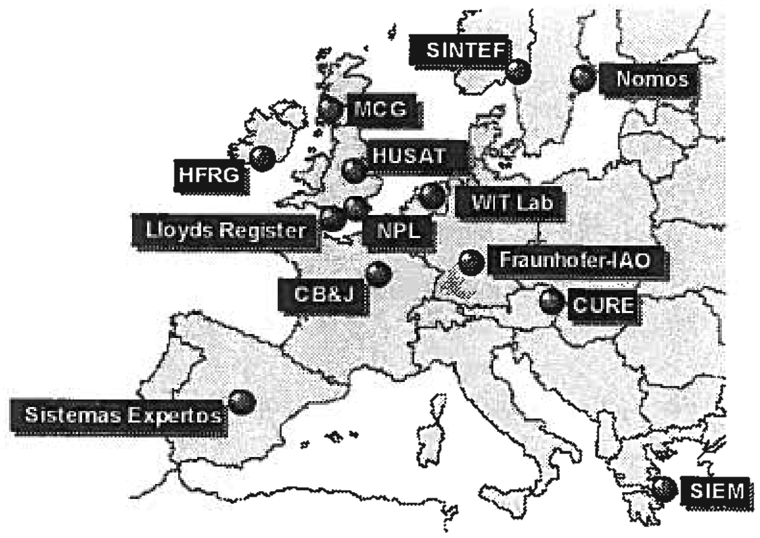


Figure 1. European Usability Support Centres

INUSE Technology Transfer Activities

NPL has contributed to the development of ISO DIS 13407 [9] which explains how to achieve usability by incorporating user centred design activities throughout the life cycle of interactive computer-based systems.

There are four user centred design activities that need to take place at all stages during a project. These are to:

- understand and specify the context of use
- specify the user and organisational requirements
- produce design solutions
- evaluate designs against requirements.

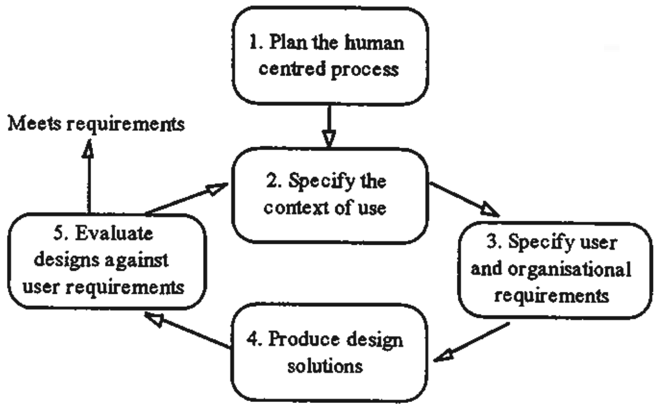

1. Plan the human centred process
Meets requirements
2. Specify the context of use
3. Specify user and organisational requirements
4. Produce design solutions
5. Evaluate designs against user requirements

Figure 2 - User centred design activities

The iterative nature of these activities is illustrated in Figure 2. The process involves iterating until the objectives are satisfied. The sequence in which these are performed and the level of effort and detail that is appropriate depends on the design environment and the stage of the design process. INUSE has produced a Handbook of User Centred Design [5] which recommends appropriate methods for implementing user centred design in different environments. This can be used by a manager to plan the user centred design process.

The INUSE project has also developed methods [6] for assessing an organisation's position on a usability maturity scale which starts with ignorance, and runs through uncertainty, awakening, enlightenment and wisdom to institutionalised usability. This can be used to provide an indication of the management commitment to implement user-centred design.

RESPECT Methods

The RESPECT project has established a structured approach to user-centred requirements engineering [14]. Techniques used for eliciting user requirements have been integrated into an iterative process of identifying the intended context of use and user requirements, developing design concepts and mockups, and testing these from a user perspective. The RESPECT process is in three phases:

1. Identify the users and stakeholders, their goals, tasks and working environment, and use this as a basis for producing and reviewing design ideas and concepts.

2. Identify and elaborate detailed scenarios of usage, and use these as a basis for developing mock ups and testing with representative users.

3. Produce detailed user requirements documenting the characteristics of each user group, their goals and tasks, and details of the expected technical, physical and organisational environment of use.

This specification can be used as a basis for elaborating and finalising the conventional technical requirements for the product. The user requirements specification forms an input to the later parts of the user centred design process documented in the Handbook of User Centred Design developed by INUSE.

International Standards

NPL has contributed to extending the principles of ISO 9241-11 into international standards for software quality, thus providing support for incorporating usability in systems design [3]. ISO DIS 14598-1 [10] and ISO CD 9126-1 [7] make quality in use (synonymous with the ISO 9241-11 definition of usability) the major quality goal in systems development [2]. ISO DIS 9241-11 describes how quality in use can be specified and measured, and ISO DIS 13407 specifies the user-centred design process which is necessary to achieve the usability and quality in use goals.

The contents are closely aligned with the methods developed by the EUSCs. This contributes to both the credibility of the methods and their technical integrity, and also enables the EUSCs to use the methods to achieve compliance with these standards, when their use is called for in a contract.

HISER Element Toolkit

NPL has recently announced availability in Europe of the HISER Element Toolkit. This is a group of tools to help organisations reduce software development costs and speed up user interface design. The toolkit provides a structured approach for involving users in the design process to enhance the usability of software and to develop rapid, repeatable processes for software development. The tools were developed by the Hiser Group in Australia and are being marketed in Europe by NPL. They provide detailed methods and procedures for practitioners to use early in the development life cycle, thus complementing the existing MUSiC tools.

There are three tool sets which include instructions and guidelines, checklists, examples, templates, and support:

- Observe and Analyse - to capture key user and usability information to set usability goals.
- Envision and Design - to capture design decisions, collaborate with users and draft prototypes.
- Evaluate and Refine - to evaluate the design against usability goals and refine designs.

References

1. Bevan N (1995a) Measuring usability as quality of use. *Journal of Software Quality*, 4, 115-130.

2. Bevan N (1995b) Usability is quality of use. In: Anzai & Ogawa (eds) *Proc. 6th International Conference on Human Computer Interaction*, July 1995. Elsevier.

3. Bevan N and Azuma M (1997) Quality in use: Incorporating human factors into the software engineering lifecycle. In: *Proceedings of the Third IEEE International Software Engineering Standards Symposium and Forum (ISESS'97)*, p169-179.

4. Bevan N and Macleod M (1994) Usability measurement in context. *Behaviour and Information Technology*, 13, 132-145.

5. Daly-Jones, O, Thomas, C, Bevan, N. (1997) *Handbook of user centred design*. National Physical Laboratory, Teddington, Middx, UK.

6. Earthy, J (1997) *Usability Maturity Model: Attitude Scale* INUSE Deliverable D5.1.4 (s). http://www.npl.co.uk/inuse

7. ISO/IEC FCD 9126-1 (1998) *Software quality characteristics and metrics - Part 1: Quality characteristics and sub-characteristics.*

8. ISO DIS 9241-11 (1996) *Ergonomic requirements for office work with visual display terminals (VDT)s - Part 11 Guidance on usability.*

9. ISO DIS 13407 (1997) *User centred design process for interactive systems.*

10. ISO/IEC 14598-1 (1998) *Information Technology - Evaluation of Software Products - Part 1 General guide.*

11. ISO/IEC PDTR 15504 (1997) *Software process assessment*

12. Kirakowski J (1996) *The Use of Questionnaire Methods for Usability Assessment.* http://www.ucc.ie/hfrg/questionnaires/sumi/sumipapp.htm

13. Macleod M, Bowden R, Bevan N and Curson I. (1997) *The MUSiC Performance Measurement Method.* Behaviour and Information Technology, 16.

14. Maguire M (1998) RESPECT User Centred Requirements Handbook. HUSAT, Loughborough.

Task Analysis in Industrial Settings: Merging Ethnography and Knowledge Elicitation

Gerrit C. van der Veer
Dept Computer Science
Vrije Universiteit, Amsterdam, the Netherlands
Gerrit@cs.vu.nl

1. Introduction

Analyzing a complex system means analyzing the world in which the system functions, or the "context of use", which comprises (according to standards like ISO 9241-11) users, tasks, equipment (hardware, software, and materials), social environment, and physical environment.
If we want to design systems for industrial settings, we need to take these different aspects of the task world into consideration. In traditional literature on task analysis from the HCI (Human-Computer Interaction) mainstream, the focus is mostly on users, tasks, and software, and the techniques aim at elicitation of expert knowledge. Design approaches for GroupWare and CSCW (Computer Supported Collaborative Work), on the other hand, often focus on analyzing the world first of all from the point of view of the (physical and social) environment, applying techniques like ethnography. In both cases, more recent developments at least include some aspects that belong to the other categories, but it still looks like one has to choose for either the one view or the other.

2. Task analysis in HCI design –collecting expert knowledge

Classical HCI features a variety of notions regarding task analysis. In many cases the design of a new system is triggered by an existing task situation. Either the current way of performing tasks is not considered optimal, or the availability of new technology is expected to allow improvement over current methods. A systematic analysis of the current situation may help formulate design requirements, and at the same time may later on allow evaluation of the design. In all cases where a "current" version of the task situation exists, it pays of to model this. Sebillotte (1988) elaborates a knowledge elicitation method to collect task knowledge and structure this into a hierarchical model of subtasks, Scapin and Pierret-Golbreich (1990) elaborate on this method and provide an object oriented formalism for modeling knowledge of existing task situations, like Sebillotte mainly focusing on activities. Task models of this type pretend to describe the situation as it can be found in real life, by asking or observing people who know the situation (e.g. Johnson, 1989). A model of the current situation is often considered of a generic nature (e.g. Sebillotte), indicating the belief of authors in this field that different expert users have at their disposal basically the same task knowledge.

All HCI task modeling is rather narrow focused, considering mainly individual people's tasks, although Johnson (1989) considers the aspect of roles and the phenomenon of allocating subtasks to different actors Most HCI approaches are based on cognitive psychology. Johnson refers to knowledge structures in long term memory. Tauber refers to "knowledge of competent users". HCI approaches focus on knowledge as can be modeled after individuals who are knowledgeable or expert in the task domain.

As a consequence of their source, HCI models seldom provide an insight in complex organizational aspects, in situational conditions for task performance, and in complex relations between tasks of individuals with different roles. Business processes and business goals are seldom part of the knowledge of individual workers, and, consequently, are seldom related to the goals and processes as found in HCI task modeling.

3. Design approaches for CSCW - Ethnography

CSCW work stresses the importance of situational aspects, group phenomena and organizational structure and procedures (Schael, 1996; Shapiro, 1996). Shapiro even goes as far as stating that HCI has failed in the case of task analysis for cooperative work situations, since generic individual knowledge of the total complex task domain does not exist. CSCW literature strongly advocates ethnographic methods.

Ethnographers study a task domain (or "community of practice") by becoming a participant observer, if possible with the status of an apprentice, being accepted as an outsider in this respect and being themselves aware of their status of analyzing observer. The ethnographer observes the world "through the eyes of the aboriginal" and at the same time is aware of his status of an outside observer whose final goal is to understand and describe for a certain purpose and a certain audience (in the case if CSCW: a design project). Ethnographers start their observation purposely without a conceptual framework regarding characteristics of task knowledge, but, instead, may choose to focus on activities, environments, people, or objects. The choice of focus is itself based on prior ethnographic observations, which illustrates the bootstrapping character of knowledge elicitation in ethno-methodology. Methods of data collection currently start with video recording of relevant phenomena (the relevance of which, again, can only be inferred from prior observation) followed by systematic transaction analysis, where inter-observer agreement serves to improve reliability of interpretation. Knowledge of individual workers in the task domain may be collected as far as it seems to be relevant, but it is in no case a priori considered the main source, and will never be considered indicative for generic task knowledge.

The ethnographic approach in unique in its attention to all relevant phenomena in the task domain that are not explicitly verbalizable by (all) experts. The approach attends to knowledge and intentions that are specific for some actors only, to conflicting goals, cultural aspects that are not perceived by the actors in the culture, temporal changes in beliefs, situational factors that are triggers or conditions for strategies, and to non-physical objects like messages, stories, signatures and symbols, of which the actors may not be aware of their functions in interaction.

4. Conceptual framework for groupware task analysis (GTA)

The framework for groupware task analysis that is presented here is based on comparing the different approaches mentioned earlier, and on an analysis of existing and proposed systems for HCI and CSCW (Van der Veer, Lenting & Bergevoet, 1996). Task models for complex situations need to be composed of different aspects. Each describes the task world from a different viewpoint, and each relates to the others. The three viewpoints (focus on agents, work, and situation, respectively) that we will apply in our approach are a superset of the main focal points in the domain of HCI as well as CSCW. Both design fields consider agents ('users' vs. 'cooperating users' or user groups) and work (activities or tasks, respectively the objectives or the goals of 'interaction' and the cooperative work). Moreover, especially CSCW stresses the situation in which technological support has to be incorporated. In HCI this is only sometimes, and then mostly implicitly, considered. In this section we will elaborate our conceptual framework.

4.1. Agents

The first aspect focuses on agents. "Agents" often indicates people, either individual or in groups. Agents are considered in relation to the task world, hence, we need to make a distinction between agents as acting individuals or systems, and the roles they play. Moreover, we need the concept of organization of agents. In situations where modern information technology is applied, actors will sometimes be non-human agents, or systems that comprise a collaboration between human agents and machine agents.

Actor
This label mostly refers to individual persons. Important for task modeling is to identify relevant types of actors, and to characterize them on relevant characteristics. Types may be identified based on two different types of variables: (1) psychological characteristics like cognitive styles or spatial ability (Van der Veer, 1990); and (2) task related characteristics like expertise or knowledge of information technology.

Role
Roles indicate classes of actors to whom certain subsets of tasks are allocated, by free choice or as the result of the organization. By definition roles are generic for the task world. More than one actor may perform the same role, and a single actor may have several roles at the same time. Roles may be performed temporarily, be negotiated between actors and accepted or refused. Actors may have internal (mental) representations of their own roles and others' roles and roles may be represented externally by instrumental or symbolic behavior and by objects (white coat, stethoscope, and wig).

Organization
'Organization' refers to the relation between actors and roles in respect to task allocation. The organization describes the agent structure in the task domain. Part of the organization is generic (as far as the structure of roles is concerned), another part concerns the current episode in the history of the task world (the organization as far as dependent on current individual actors and the roles they currently perform). Delegation and mandating responsibilities from one role to another are part of the organization, as is the way roles are allocated to actors. In organizational structure roles can be hierarchically related in several ways: a role can be a subtype of another role (a sales manager is a manager), or roles may be part of a role (a nurse is part of the company health department, which is part of the personnel division).

4.2. Work

Some approaches refer to goals as the unit of description of work (GOMS: Card, Moran and Newell, 1983), but we prefer to focus on the structural as well as dynamic aspect of work, hence, we will take 'task' as the basic concept, and 'goal' as an attribute. The concepts of task and goal in most frameworks have either a many to one or a one to one relation – several tasks may have the same goal, and each task has exactly one goal. In activity theory tasks are referred to as 'actions' (which are, like in HCI task analysis approaches, considered to be hierarchically structured), where long-term tasks are referred to as 'object' or 'motive' (Nardi, 1995). We make a distinction between tasks and actions in the 'classical' HCI terminology, and, moreover, we will elaborate task structure and the structure-related concepts of protocol and strategy.

Task
Tasks can be identified at various levels of complexity. The unit level of tasks needs special attention. Payne and Green (1989) call this the 'simple task', but this notion may either indicate an artifact of a system, or a psychological concept, which sometimes results in ambiguity in analysis. We need to make a distinction between (1) the lowest task level that people want to consider in referring to their work, the 'unit task' (Card, Moran, and Newell, 1983); and (2) the unit level of task delegation that is defined by the tool that is used in performing work, like a single command in command driven computer applications. This last type of task we will call 'Basic task' (Tauber, 1990). Unit tasks will often be role-related.

Complex tasks may be split up between actors or roles. Unit tasks and basic tasks may be decomposed further into (user) actions and (system) events, but these cannot really be understood without a frame of reference created by the corresponding task, i.e., actions derive their meaning from the task.

Task structure
The task structure will often at least partially be hierarchical. For the indication of temporal order and dependency structure, concepts like the 'constructors' of Scapin and Pierret-Golbreich (1990) are relevant. Single actors do not always know task structures, mainly when different roles are involved in performing different subtasks. On the other hand, performance on certain subtasks may influence the procedures for other subtasks.

Actions
Actions are identifiable components of basic tasks or unit tasks, which have a meaning in performing a unit of work, but which derive their meaning only from the task they are part of. For instance hitting a return key has a different meaning depending on whether it concludes a command, or confirms the specification of a numerical input value. The speech act of confirmation has a different meaning depending on whether it follows another person's question or command. On the other hand, actions are the smallest elements of a basic or unit task that change or define the meaning of that task. In describing actions, the goal is to identify the meaning, not the physical characteristics.

In Activity theory these components seem to be equivalent to 'operations', which are at the level of automatism and the elements of subconscious feed-back loops. This theory stresses the phenomenon that actions may become operations by continued learning and experience and that they must become 'actions' again when the operations are frustrated. Typical actions in HCI and CSCW are the specification of objects or events, and speech acts. Actions may aim at changing (or operating on) attributes, 'location' or existence of objects, change attributes of the environment, or may effect mutual task performance between different actors. Actions that concern the 'content' of an object may often be considered to act on other objects that are contained in the current object ('themes', see below). Actions, as parts of basic tasks or unit tasks, are often not explicitly 'known' (i.e., verbalizable) or actors are reluctant or unable to be very precise in this respect.

Protocols
This concept indicates actual 'rules' as turn out to be applied for decomposing tasks, to be distinguished from 'rules' that may be stated explicitly in instructions which are sometimes not

actually followed. Protocols may be *situated*; i.e., the environment and the presence of actors with certain roles may constitute conditions for protocols to be triggered.

Strategies
'Strategies' indicate structures that can be considered protocols used mainly by experts or typically preferred by them. These structures will often be situated in the same way as protocols are. Strategies may have started from explicit problem solving and knowledge formation episodes and subsequently have become implicit expert knowledge. Strategies will be role related.

4.3. Situation

Analyzing a task world from the viewpoint of the situation means detecting and describing the environment (physical, conceptual, and social) and the objects in the environment. Object description includes an analysis of the object structure.

Object
Each thing that is relevant to the work in a certain situation is an object in the sense of task analysis. In this framework, 'objects' are not defined in the sense of 'object oriented' methods. Objects may be physical things, or conceptual (non-material) things like messages, gestures, passwords, stories, or signatures. Non-material objects as well as physical objects may in the task situation be referred to by external representations of different character: verbal labels, graphics, metaphors, gestures. Actors that perform a certain role may be objects in a task situation and will be labeled 'active objects'. Non-human system components like computer based agents may also be active objects.
The identification of relevant objects will depend on the condition of knowledge (explicit or implicit) and on whether the object figures in a task for a single person or in group situations. Relevant objects may be used to transport meaning and information between different agents without any of them being aware of the objects' nature (e.g., anecdotes that contain strategic information). As far as explicit knowledge is involved, analysis of verbal material from archival sources or from interviews may be of help, starting with the identification of nouns in relation to task references. For implicit knowledge about objects, observations and ethnographic methods have to be used, both for detection and for description.

Object structure
In order to describe the semantics of objects, two kinds of relations between object types have to be identified.
1. Object types are related via a type hierarchy, indicating sub-type - super-type relations. Sub-types inherit the characteristics of their super-type as far as no further specifications have to be added. Analysis will reveal the exact relations of object types of certain levels in a type hierarchy featuring in the task world.
2. Semantic relations between object types may metaphorically be indicated by place relations, (Tauber, 1990, uses the concept 'theme' for this relation in his ETAG formalism), and where objects may 'move' from one place to another (each place being provided by an object). Apart from the relation between object types, objects will be related to tasks as agent (active objects), as subject, or as featuring in conditions of task structures. The identification of object structures will be an analytic (HCI type) activity, based on verbal protocols from actors and on systematic observation of the situational relations in which objects are used.

Environment
The task environment is the current situation for the performance of a certain task. It includes actors with roles, conditions for task performance and for strategies and protocols, relevant objects, and artifacts like information technology that are available for subtask delegation. The history and temporal structure of relevant events in the task situation is part of the actual environment. The environment features as condition for task structures (inclusive protocols and strategies as far as these are situated). The analysis and description of environments often will need ethnographic methods.

5. Conclusion: Merging task knowledge

In order to analyze and represent task knowledge for complex industrial settings, we need both the points of view of classical HCI, and the type of insights gained in CSCW approaches. Consequently, we need a conceptual framework like the one proposed here. The techniques for collecting the relevant task knowledge will be a combination of the knowledge elicitation, observation techniques, and ethnographic techniques. Based on the resulting knowledge from these techniques, models can be developed that allow a representation along the different views mentioned in the conceptual framework. GTA offers an approach derived from both CHI and CSCW. It provides the basic concepts and views, a way to combine techniques for knowledge collection and representation, as well as tools for modeling and formalizing design decisions.

References

Card S K & Moran T P & Newell A 1983 'The Psychology of Human-Computer Interaction', Hillsdale, NJ, Lawrence Erlbaum Ass

ISO DIS 9241-11 1996 'Ergonomic requirements for office work with visual display terminals (VDTs): - Part 11: Guidance on Usability', ISO

Johnson P 1989 'Supporting system design by analyzing current task knowledge' in D Diaper (ed) *Task analysis for human-computer interaction*, Chichester, Ellis Horwood

Nardi B (ed) 1995 'Context and Consciousness: Activity Theory and Human Computer Interaction', Cambridge MA, MIT Press

Payne S J & Green T R G 1989 'Task-Action Grammar: the model and its developments' in D Diaper (ed) *Task analysis for human-computer interaction* Chichester, Ellis Horwood, pp 75-105

Scapin D & Pierret-Golbreich C 1989 'Towards a method for Task Description: MAD' in L Berlinguet & D Berthelette (eds) *Work with display units 89*, Amsterdam, Elsevier

Schael T 1996 'Information systems in public administration: from transaction processing to computer supported cooperative work' in D Shapiro, M J Tauber & R Traunmueller (eds) The design of computer supported cooperative work and groupware systems, Amsterdam, North-Holland

Sebillotte S 1988 'Hierarchical planning as a method for task-analysis: the example of office task analysis' in Behaviour and Information Technology 7 (3), pp 275-293.

Shapiro D 1996 'Ferrets in a sack? Ethnographic studies and task analysis in CSCW' in D Shaspiro, M J Tauber & R Traunmueller (eds) The design of computer supported cooperative work and groupware systems, Amsterdam, North-Holland

Tauber M J 1990 'ETAG: Extended Task Action Grammar - a language for the description of the user's task language in D Diaper & D Gilmore & G Cockton & B Shackel (eds) *Proceedings INTERACT "90*, Amsterdam, Elsevier

Van der Veer G C 1990 'Human-computer interaction: learning, individual differences, and design recommendations' PhD Dissertation, Amsterdam, Vrije Universiteit

Van der Veer G C, Lenting B F & Bergevoet B A J 1996 'GTA: Groupware Task Analysis - Modeling Complexity' in Acta Psychologica 91, 297-322.

Evaluation Methodology – Its Goal and Benefits

Reinhard Oppermann
Oppermann@gmd.de
German National Research Centre for Computer Science (GMD)
Institute for Applied Information Technology FIT
D-53754 Sankt Augustin

1. Importance of Evaluation

There is an increasing need for practical and comprehensive evaluation methods and tools for software ergonomic evaluation. This holds for the design of systems to get early input for usability refinements (formative evaluation) and for the final check of a given application for both the manufacturer and for the employer (summative evaluation). There are different classes of methodologies to perform an evaluation. None of them is *the* evaluation method. Most suitable is a combination of different methods tailored to the specific needs of evaluation rational. Real or representative users are essential for any evaluation. But user studies are time consuming and expensive. Expert based assessment can be performed for aspects of the evaluation that are objective enough to be described by evaluation items.

Evaluation criteria are described in the literature, in industrial guidelines and in standards agreed upon in national or international standard organisations. I want to present and discuss a methodology that supports software evaluation according to ISO 9241 which is an important ergonomic standard being still under development.

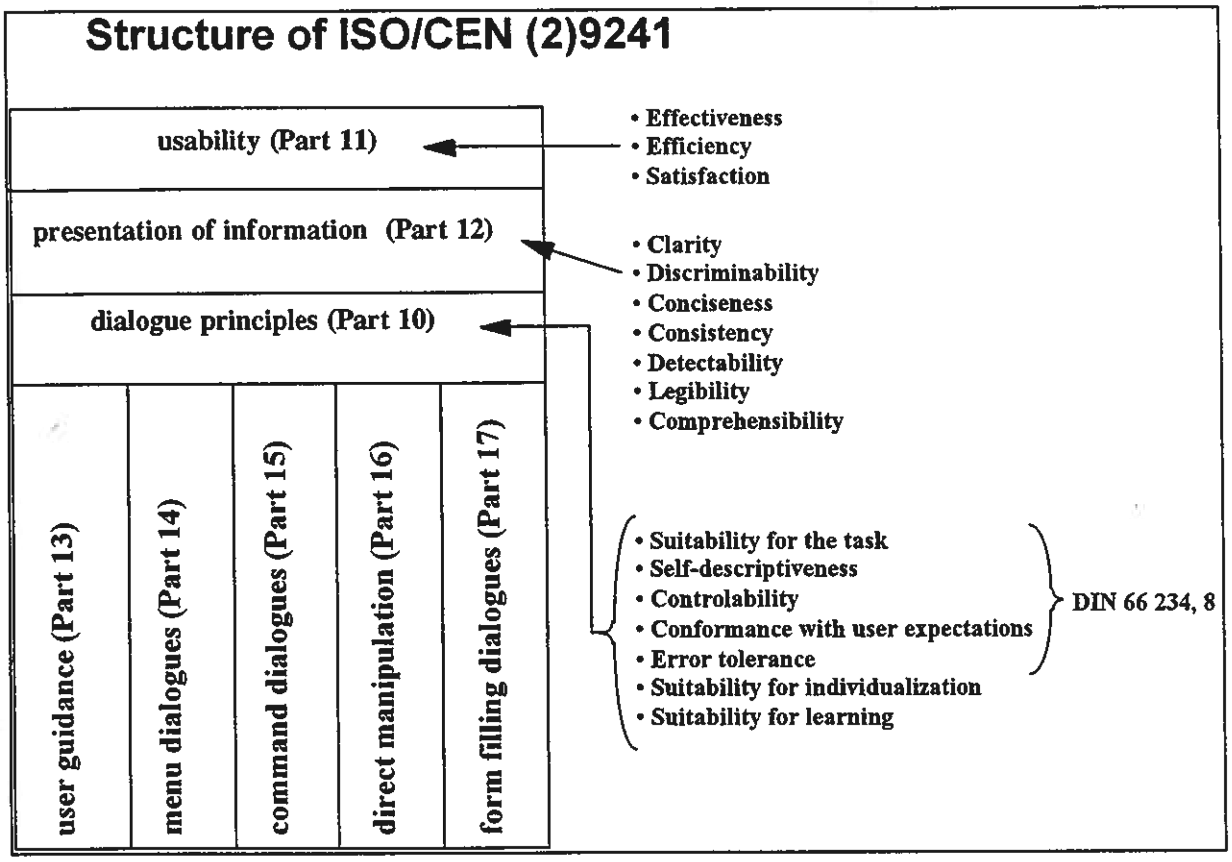
2. ISO 9241

Software-ergonomic evaluation is aimed at assessing a system's degree of usability. Standards are based on empirical evidence and practical and economical feasibility. Not all desiderata from research are accepted as standards. For example, there may be objections due to technical or financial constraints.

In some cases, the application of standards is enforced by law. The European Union (EU), for example, published the directive 90/270/EWG concerning the minimum safety and health requirements for VDT workers (EEC 1990) to establish common working conditions for users of visual display terminals. The national governments participating in the EU have transformed this directive into national law. The international standardisation activities of ISO 9241 concerning ergonomic requirements for visual display terminals form the basis which define the relevant technological requirements necessary to fulfil the directive.

ISO 9241, the "Ergonomic requirements for office work with visual display terminals (VDTs)", is far from being a pure technical standard that can be assured by quantitative measures. Rather, it requires interpretation and tailoring to be useful in user interface evaluation and reflection about the state-of-the-art technology in research and development. It is subject to an ongoing process of discussion and bargaining, and it has to successfully pass through several stages of negotiation and acceptance. Different expertise and interest influence the results, and they establish a "minimum level of user oriented quality" (Dzida 1995).

ISO 9241 consists of eight parts to be considered during a software-ergonomic evaluation (the other parts are dedicated to task and hardware issues). The relevant parts of the software-ergonomic standard are shown in Fig. 1 (see Dzida 1995, 94 for the general idea of the structure).



Structure of ISO/CEN (2)9241
usability (Part 11)
presentation of information (Part 12)
dialogue principles (Part 10)
user guidance (Part 13)
menu dialogues (Part 14)
command dialogues (Part 15)
direct manipulation (Part 16)
form filling dialogues (Part 17)
• Effectiveness
• Efficiency
• Satisfaction
• Clarity
• Discriminability
• Conciseness
• Consistency
• Detectability
• Legibility
• Comprehensibility
• Suitability for the task
• Self-descriptiveness
• Controlability
• Conformance with user expectations
• Error tolerance
• Suitability for individualization
• Suitability for learning
DIN 66 234, 8

Figure 1: Structure of the multi-party standard ISO 9241

The *concept of usability* is defined in part 11 by effectiveness, efficiency, and satisfaction of the user.
The *dialogue requirements* are described in part 10 by the suitability of the task, self-descriptiveness, controllability, conformance with user expectations, error tolerance, suitability for individualisation, and suitability for learning. Part 10 establishes a framework of ergonomic "principles" for the dialogue techniques with high-level definitions but with only illustrative applications and examples of the principles. The principles of the dialogue represent the dynamic aspects of the interface and can be mostly regarded as the "feel" of the interface.
The *information presentation* is described in part 12 by the "attributes" clarity, discriminability, conciseness, consistency, detectability, legibility, and comprehensibility. The "attributes of presented information" represent the static aspects of the interface and can be generally regarded as the "look" of the interface.
Requirements for *user guidance* (prompts, feedback, status, error support and help facilities) are described in part 13. The application of the user guidance rules also contribute to the application of the dialogue principles.

The requirements for user guidance and several dialogue techniques are described in part 13 to 17. Each of the recommendations given in part 13 to 17 contributes to at least one of the dialogue principles and to the attributes of information presentation, but this relationship is not explicitly stated. Part 13 to part 17 of the standard define more or less low-level and fairly exhaustive requirements for the user interface. In many cases task, user, and technical environment aspects are considered as conditions of the applicability or relative relevance of the specified requirements. These aspects constitute the context of use defined in part 11 to be considered when applying the standard to a given work system.

3. ISO 9241 Evaluator - An Evaluation Procedure

The ISO 9241-Evaluator supports the conformance test of a given application with ISO-9241, part 10 to 17. The evaluation procedure is an approach being designed to support the ex-post evaluation of a user interface that is now open to being used for evaluations of interfaces under development (mock-ups, prototypes etc.).

The ISO 9241-Evaluator is a guideline oriented expert-based evaluation method that prepares the requirements of the multi-party standard ISO 9241 to be tested in 546 test items. The contribution of the ISO 9241 Evaluator can be seen in the software support for the process of an evaluation. With the ISO 9241 Evaluator, the support for the testing, the documentation of the testing, the evaluation, and the report of the results is provided.

The ISO 9241 Evaluator is a subjective evaluation method because the expert examines and answers questions according to his personal assessment. However, it is also objective because the ergonomic requirements are operationalised and precisely formulated, thus enabling the evaluator to answer questions based on clear test rules and traceable conditions. The evaluator is a human factors expert using the methods to evaluate the conformance with ISO 9241. The expert approach is based less on a task to be performed by the targeted system than on questions asked by software ergonomics. Advantages of an expert based evaluation method are: relatively fast, uses few resources, provides an integrated view, and can be addressed to a wide range of behaviour.

There are several elements of a test item:
The test instruction gives the evaluator information about how to reflect the context of use for the given application, i.e., if and which user characteristics, task characteristics and environmental characteristics have to by considered. The comment contains desirable ergonomic requirements. The answers to the test item in the particular test situation are made by ticking off one or more of the check boxes representing several aspects of the test item. All answer options represent the complete ergonomic requirements of this test item, as they have been defined in the corresponding part of the ISO 9241. Test items have to be answered in particular test situations, typically more than once. Each test situation will be shortly characterised by the evaluator (e.g., menu structure of the main menu; menu structure of the customer window). With the help of integrated capture tools (Lotus ScreenCam and Microsoft Paintbrush), all on-screen behaviour and window sequences, together with verbal and textual comments of the evaluator can be captured. The logged examples can be used by the evaluator to explain detected deficiencies. This is very helpful for communication with the software designers, because the evaluator can explain the deficiencies in real examples.

For each test situation, a new rating of the achievement of the ISO 9241 requirements will be done automatically by the software. Answering a test item in different test situations is necessary for a comprehensive evaluation of a software product. Different answers in different situations can be an indication of a context sensitive adaptation of the interface to different contexts of use, but they can also be an indication of an inconsistent user interface. The evaluator's description of the test situations allows him to distinguish between appropriate adaptation or inconsistency in the interpretative phase of the results, and it allows for reconstruction of the given answers in a redesign discussion with the designer at a later time.

The definition and documentation of test situations allow the evaluation procedure to be used not only for summative but also for formative evaluation purposes. It can be embedded in the software engineering process at different stages (see Reiterer and Oppermann 1993; Oppermann and Reiterer 1997). Hix (1995) discusses in greater detail the incorporation of usability evaluation techniques into the software lifecycle.

4. Summary

Many different evaluation methods are currently available. The choice of a method for a specific evaluation depends upon the stage of system development, the kind of user involvement, the type of data necessary for the kind of results required, the available expertise, the possible place of the evaluation and the available resources (time and money). There is an increasing need for practical and comprehensive evaluation methods and tools for conformance testing with ISO standards. Practical means that the amount of time and resources must be manageable in software projects. Comprehensive means that the context of use has to be considered during the evaluation of user interfaces. For a comprehensive evaluation of usability, it is necessary to use more than one evaluation method. For example, an expert method for diagnosis of general usability principles in a user interface, combined with the use of a subjective method to evaluate the user interface to show the users' level of satisfaction. Another example may be an expert method for quick diagnosis of specific usability problems in a prototype, combined with the use of an objective method to measure the redefined prototype to evaluate the user performance with the help of representative test tasks.

The ISO 9241-Evaluator presented in this article is a practical expert-based evaluation method that can be integrated into a comprehensive evaluation approach. In particular, it takes the context of use into consideration and provides extensive computer support for the use of the evaluation procedure.

References

Dzida, W. 1995, Standards for user-interfaces, Computer Standards & Interfaces 17 (1995), 89 - 97

EEC 1990, The minimum safety and health requirements for work with display screen equipment, Directive of the European Economic Community, 90/270/EEC

Hix, D. 1995, Usability Evaluation: How Does It Relate to Software Engineering? in: Y. Anzai, K. Ogawa, and H. Mori (eds) Symbiosis of Human and Artifact. Amsterdam: Elsevier Science B. V., pp. 355 - 360.

IBM 1991, Systems Application Architecture, Common User Access, Advanced Interface Design Reference.

ISO 9241 1994, Ergonomic Requirements for Office Work with Visual Display Terminals, Part 8, Requirements for displayed colours, Draft International Standard, 1994.

ISO 9241 1994-1998, Ergonomic Requirements for Office Work with Visual Display Terminals, Int. Standard.

ISO 9241 1994, Ergonomic Requirements for Office Work with Visual Display Terminals, Part 17, Form filling dialogues, Committee Draft, 1994.

Oppermann, R. H. Reiterer 1997, Software Evaluation using the 9241 Evaluator. Behaviour & Information Technology 16 (1997), 4/5, 232 - 245.

Reiterer, H., and R. Oppermann 1993, Evaluation of User Interfaces, EVADIS II - A comprehensive Evaluation Approach, Behaviour & Information Technology 12 (1993) , 3, 137 - 148.

Usability Contracts and Other Strategies of Usability Transfer in Industrial Settings

Paul Hubert Vossen *Paulus.Vossen@iao.fhg.de*
Fraunhofer- IAO, Stuttgart, Germany

Preliminary Problem and Stakeholder Analysis

What does industry - system, software and product engineers - need in order to accept and handle the additional burdon put on them when working in accordance with the results of human-computer-interaction research or the tenets of usability engineerin? Conversely: what do we as usability professionals or human factors experts need to get an impressive, substantive and lasting influence on product development projects, i.e. on the development process itself as well as on the resulting interactive products — information and communication systems, and other devices and equipment with embedded interactive software for the professional or consumer market?

And what about the other stakeholders whose special needs or wishes regarding product usability shall be taken into account: *marketing people*, who will be happy, because the product will be well accepted by its intended users and indeed more so than similar products from the competitors; *sales department*, who will be happy, because the product is sold in large quantities; *management and financial departments*, who will be happy, because the business goals will be met within time and budget constraints; *system developers*, who will be happy, because they have been able to concentrate more on some very exciting architectural concepts using latest programming methods; *product designers*, who will be happy, because they have been able to push forward latest (aesthetic) design concepts; and - last but no least - the *users* or *consumers*, who will be happy, because they immediately understand the myriad of functions and features embedded in the product (even the ones they did not ask for ...), need little time to learn to operate it without any error or omission, are extremely fast with it and enjoy the intrinsic appeal of the product.

From Software Crisis to Usability Crisis

In hindsight, the discussion started off in the 70's with the now muted „software crisis" — only few people like David Parnas dare to say loud and clearly, that even today there does not yet exist such a thing as „software engineering" comparable in professional status and standing to the older engineering disciplines; I wonder what he would say about „usability engineering" ... — and then was taken up by psychologists, ergonomists and - most recently - product designers who claimed to have the knowledge and know-how to substantially improve the practices and results of software developers and - more generally - product developers, thereby resolving the now aptly called „usability crisis".

Is there really a Usability Crisis? The consequence of not taking human performance or - more generally - usability requirements into account has already been formulated by Bailey as early as 1982 (p. 221):

"If we do not clearly state the requirements from the beginning we cannot expect human performance considerations to be taken seriously. In fact, in the absence of human performance requirements, other system requirements will take precedence — those related to software or hardware development. When this happens the entire development process will focus on meeting these other requirements. And when the system is operational, people will be left to perform as best they can without adequate provision for ensuring an acceptable level of human performance."

Eight years later Chapanis and Budurka summarise the deplorable state of affairs with respect to human factors in user interface design as follows (pp. 479-482):

- Human factors is not part of main stream engineering
- Human factors has no binding way to influence development
- Human factors professionals focus too narrowly on human requirements
- Human factors decisions are left to (interface/interaction) designers
- Present guidelines and standards are too general
- Present system, hardware and software specifications are not specific about HCI requirements
- Existing guidelines and standards are not testable
- Usability evaluation occurs too late in the development cycle

And again, eight years later the situation has not dramatically improved, although of course we have gained a lot of experience with all kinds of usability transfer models in the meantime. The question arises where the flow of information is disrupted along the chain of human factors, i.e. psychological and ergonomic, research and practice on the one hand and final product development and delivery on the other hand. What is the reason that designers of user interfaces to all kinds of electronic information and communication systems are only partially capable to include long established and widely accepted human factors principles in their products?

„We did it our way ...“. During the last 20 years or so, a whole arsenal of potentially promising solutions has been proposed and introduced in diverse organisational and industrial settings (cf. e.g. Wiklund 1994). Promisses, no proofs! Insofar as there has been an objective evaluation of their impacts versus requirements or benefits versus costs on the basis of stated criteria, the reported successes seem to be specific for the organisation or project. There exists no empirical evidence for the transferabilty of any one approach and its success to other organisational and industrial contexts. Didn't we yet find the correct solution or are we just overlooking the real problem, thus solving the wrong one?

Framing the Usability Problem. Remarkably enough (caused by economic pressures from within industrial reality?) the different approaches proposed by allegedly empirically minded researchers have been and often still are not so empirically grounded as one might expect. Rather what happens seems to be the following: acknowledging the many aspects and facets of the (assuming there is only one) *usability transfer problem*, one particular framing of it is put forward, leading to the formulation of a particular *usability transfer model*, which is then enthousiastically proposed or reported as the (again: assuming there is only one) solution to most if not all evils - at least within the organisation(s) in which it was put into practice. Some of those usability transfer models have become so popular that they have got a name reminiscent of its context of „discovery“, e.g. the participatory model of the Scandinavian School.

Ways out of the Usability Crisis

Let us therefore make up a simple taxonomy of those different approaches to the *usability crisis*, not bothering too much about completeness or other quality criteria for such taxonomies, in order to get a feeling for the hugh diversity of offered solutions and to set the context for a discussion of one neglected usability transfer model — at the risk of being accused of adopting that same non-empirical approach criticized above ... It is neither our goal nor does space allow us here to describe and criticize each of these approaches in detail. Some of their strengths and weaknesses are obvious enough, even on first thought. It is tempting therefore to suggest to adopt not a single approach, but a kind of integrated approach, i.e. taking the best of each single one in one big framework. But which practitioner would really have the money, time and patience to work that out for each of his projects ...? There is really not a simple solution!

Process-based approaches. Usability is optimally transferred, it is assumed and claimed, by adopting some appropriate development process, fully specified and agreed among all participants. Emphasis is here on working out the right sequences and combinations of activities, probably in the form of a project scheme. So-called quality management models may be subsumed under this label. Models which specify test procedures also belong to this category.

People-based approaches. Usability is optimally transferred, it is assumed and claimed, by having or hiring the right people for the job to be done. Emphasis is on existing knowledge and know-how of individuals and their co-operation in teams. Ask a consultant, put an expert in the team, hire an usability professional. It is assumed, that all necessary knowledge and know-how has already been acquired by those people during previous education, training or industrial experience. Any missing bits of wisdom will be acquired on the fly. Maturity models belong here.

Training-based approaches. Usability is optimally transferred, it is assumed and claimed, by transmitting any missing knowledge and know-how through dedicated - possibly ad hoc - seminars, workshops (like this one?), tutorials, and by offering and using specific text- and handbooks, manuals, etc. The emphasis is on one-shot delivery of the lacking information. The basic assumption is, that the required knowledge is of a generic nature and can easily be codified and presented for off-line learning in advance of any concrte development project.

Document-based approaches. Usability is optimally transferred, it is assumed and claimed, through documents in one or more stages of a development project. The documents may specify usage requirements, or may propose specific design solutions, or may specify a test and evaluation plan, or may describe the results of those tests and evaluations and make formal recommendations. Emphasis is on how such documents should look like and how they will be processed during system development (some implicit assumptions about peopel and processes have to be made...). This class includes such models which rely on standardized specification documents, e.g. using document templates (predefined tables of contents, perhaps with embedded instructions how to expand the outline in concrete text). It also includes the use of proprietary, national or international standards.

There are still other approaches, e.g. establishing a professional organisation and code of practice for usability practitioners or the establishment of all kinds of supporting institutions like (inter)national conferences or the EUSC network, but in view of limited space we will not further elaborate on this theme. The general idea will be clear by now.

Usability Contracts

I would like to conclude this paper with a short discussion of a neglected aspect of any form of cooperation between usability professionals (syn.: human factor experts) and product designers (including here: software developers) in the context of a concrete development project: *the usability contract* (in analogy to social contract). Too often, in discussions and proposals for setting up or enhancing such cooperation, it is assumed as self-evident, that:

(1) most people involved are intrinsically motivated to cooperate in a most effective and efficient way and that
(2) most often they will succeed in that just by good will and the provision of some methodological recommendations and good practice guidelines.

In my view this is a rather naive assumption merely based on wishful thinking, not on empirical evidence, and actually in sharp contrast to established practices in engineering, including software engineering (if it exists at all ...). The temptation is strong to blame the people involved or at least their insufficient education and training in *human factors* methods and *human-computer-interaction* issues or their lack of experience with working in projecs, or to blame the particular methods provided or at least the form in which those methods have been offered, e.g. in a form not suitable for industrial contexts with often strict and rather limited budgets and time horizons.

Now there is of course nothing wrong with improving (software) engineering curricula so as to include more of ergonomic and design knowledge and know-how (cf. initiatives in Germany or USA) or to think about more clever ways of disclosing and accessing such ergonomic wisdom for designers and developers, e.g. by exploiting existing powerful technologies like hypertext, multimedia, CD-ROM, WWW, etc. However these approaches circumvent an important problematic aspect of any large development project: such a project does not only have technical and marketing goals to achieve, it also has social dimensions and legal aspects to fulfil.

Chapanis and Budurka (1990) propose the *human-computer interface requirements document* as a kind of binding contract (see their list of diagnostics, page 2, this paper) between the human factors (or usability) experts and the software developers on a particular project. This document, in which all user interface design features will be specified, has a status similar to that of a conventional software or system specification document, which it complements in a natural way. In their paper they also gave various examples (in the form of templates, as a form of memory aid and standard) of the kind of documents that would fit sich a description (not included here for space reasons). The document is the result of a translation process which I call usability mapping. A review of the existing literature on both sides of the user interface research community, i.e. software requirements engineering and human-computer interaction research furthermore revealed that the very craft of *usability mapping* or *usability-driven user interface specification* is either completely skipped, only mentioned in passing, or inadequately narrowed down to a technique or development phase which happens to suit the particular author or researcher (cf. Davis 1993, Macauley 1996, Sommerville 1996, Helander et al. 1997, Carroll 1997).

Formal characteristics of user interface specifications. In an ideal universe, we would not need this kind of explicit design documentation guidance, however, for the time being, given the fact that both human-computer-interaction professionals and user interface software designers have to live in (at least) two conceptual worlds simultaneously, it is best to be explicit about the way that good user interface specifications should be written and not to leave it to the intuition and imagination of either side to figure that out in each single project. The formal characteristics of a user interface specification, providing a useful checklist to guide the process of documenting user requirements, are, among others: non-ambiguity, completeness, verifiability, Consistency, Modifiability, Traceability.

References

Bailey R. W. (1982). Human performance engineering: a guide for system designers. Englewood Cliffs, N.J.: Prentice-Hall

Carroll J.M. (ed.) (1997). Human-Computer Interaction. Part VII of The Computer Science and Engineering Handbook, Allen B. Tucker Jr. (Ed.), Boca Raton: CRC Press, Inc.

Chapanis A. & Budurka W. J. (1990). Specifying Human-Computer Requirements. In: Behaviour & Information Technology, vol. 9, pp. 479-492

Davis A. M. (1993). Software requirements: objects, functions, and states. London:Prentice Hall International, Inc.

Helander M., Landauer T. K. & Prabhu P. V. (eds.) (1997). Handbook of Human-Computer Interaction. Amsterdam: North-Holland (Second, completely revised edition)

Macauley L.A. (1996). Requirements Engineering. London. Springer.

Sommerville I. (1996). Software Engineering. Wokingham: Addison-Wesley.

Wiklund M.E. (1994). Usability in practice: how companies develop user-friendly products. Boston: AP Professional.

Moderation Instead of Modelling: Some Arguments against Formal Engineering Methods

Matthias Rauterberg
rauterberg@ipo.tue.nl
Center for Research on User-System Interaction (IPO)
Den Dolech 2, NL5600 MB Eindhoven, The Netherlands

Abstract. The more formal the used engineering techniques are, the less non-technical facts can be captured. Several business process reengineering and software development projects fail, because the project management concentrates too much on formal methods and modelling approaches. A successful change of work and organisational processes and structures needs primarily the agreement among all involved groups. To come up with a shared understanding, the quality and the quantity of communication are critical success factors. The moderation instead of modelling of human activities is of crucial importance. To change an organisation means to persuade people to behave in a different way than before. A model per se does not change anything, only an agreement among all affected parties is the basis of a real change. Participatory design and the adequate moderation of all necessary learning processes are the crucial success factors.

1. Introduction

The more or less formal methodology of 'business process reengineering' (BPR, Hammer and Champy 1993) has been discussed intensively in the last few years. By means of a fundamental redesign of business processes the methodology of BPR wanted to achieve an increase in efficiency of 20 to 40 per cent. As we know today most of all BPR projects in USA failed (Womack 1995). If BPR intends to be more than just 'the latest thing', its fundamentals have to be carefully analysed. A recourse to already known concepts shows, that some connected problems have to be solved to change

BPR from 'the latest thing' into real business revolution. What are the reasons for the unsuccessfulness of BPR: Is it the distinction between 'formal' versus 'informal' approaches, or are there other reasons? We will discuss this problem in the context of software development and the implementation of modern technology (e.g., information systems) in enterprises. The most important reason that we can do this, is the fact, that the implementation of modern technology in companies will nearly always affect the work and organisational structure of this company.

Software design is work design, and work redesign deals always with humans! This aspect is one of the 'new' insights of Champy (1995). Analysis of current software development processes brings to light a series of weaknesses and problems, the sources of which lie in the theoretical concepts applied, the traditional procedures followed (especially project management) as well as in the use of inadequate formal design methodologies. DeMarco and Lister (1987) reported that fifty per cent of all investigated projects were stopped, failed, or delivered a product that was never used; the larger the project, the greater the chance to fail! In nearly all of these projects the critical factor for success or failure was not the used technology. But, what are the relevant success factors?

A possible answer points to the significance of the 'human factor'. The analysis of actual software development processes shows that there are three essential barriers: the specification barrier, the communication barrier and the optimisation barrier. Speaking quite generally, one of the most important problems lies in coming to a shared understanding by all the affected people of the component of the work system to be automated (Naur 1985) that is, to find the answers to the questions of 'if', 'where' and 'how' for the planned implementation of technology, to which a shared commitment can be reached. This involves, in particular, determining all the characteristics of the work system that are to be planned anew. Every work system comprises a social and a technical subsystem. An optimal total system must integrate both simultaneously. To arrive at the optimal design for the total enterprise, it is of paramount importance to regard the social subsystem as a system in its own right, endowed with its own specific characteristics and conditions, and a system to be optimised when coupled with the technical subsystem.

2. Barriers in the Framework of Traditional System Development

There is an increasing tendency in information system development to consider not only the functionality that should be automated, but to pay attention to the work and organisational structures around the information system. Simple, intuitive, and powerful techniques are needed to model all relevant parts of the environment in the requirement analysis phase.

Contemporary modelling techniques are either (a) weak in expression or (b) cluttered with rigorous details. In case (a) models become too vague to be meaningful, while in case (b) the techniques make modelling of rather simple dynamic systems a complex task and hardly facilitate communication with users.

Today the object-oriented approach is becoming increasingly popular. Object-oriented analysis (OOA), object-oriented design (OOD) and object-oriented programming (OOP) are well established (Booch 1994, Brunet et al. 1994). Some of the most often stated advantages of OOA are: (a) object orientation corresponds very well to the human perception of an organisation (it is a 'natural' view), and (b) choosing an object-oriented structure also for the analysis, the system developer achieves a smooth transition to design and implementation.

The main critic against advantage
(a) is that processes are more fundamental than things or objects (e.g. it is difficult to specify declarative business rules within an object-oriented structure), and against
(b) that a small gap can be achieved only by pulling analysis to design (this can be done for example with a requirement engineering approach that is problem and not target oriented).

BPR projects with their primary focus on the processes should be more successful than OOA projects. But this statement seems to be not true. Why? The specification barrier is an immediate problem even at a cursory glance. How can the software developer ascertain that the client is able to specify the reqüirements for the subsystem to be developed in a complete and accurate way that will not be modified while the project is being carried out? The more formal and detailed the medium used by the client to formulate requirements, the easier it is for the software developer to incorporate these into an appropriate software system. But this presumes that the client has command of a certain amount of expertise. However, the client is not prepared to acquire this or perhaps is in part not in a position to do so before the beginning of the software development process. It is therefore necessary to find and implement other ways and means, using from informal through semiformal to formal specification methods (see Pohl

1993). It would be a grave error with dire consequences to assume that client susually people from the middle and upper management level are able to provide pertinent and adequate information on all requirements for an interactive software system. As a result, the following different perspectives must be taken into consideration in the requirement phases.

The communications barrier between applier, user and end user on the one hand and the software developer on the other is essentially due to the fact that 'technical intelligence' is only inadequately imbedded in the social, historical and [micro] political contexts of technological development. Communication between those involved in the development process can allow non-technical facts to slip through the conceptual net of specialised technical language, which therefore restricts the social character of the technology to the functional and instrumental. The more formal the used techniques are, the less non-technical facts can be captured. The application oriented jargon of the user flounders on the technical jargon of the developer. This 'gap' can only be bridged to a limited extent by purely linguistic means, because the fact that their respective semantics are conceptually bound makes the ideas applied insufficiently precise. Overcoming this fuzziness requires creating jointly experienced, perceptually shared contexts. Schüler and Namioka (1993) describe several approaches how to do this. Beyond verbal communication, visual means are the ones best suited to this purpose. The stronger the perceptual experience one has of the semantic context of the other, the easier it is to overcome the communications barrier (Ehn and Kyng 1991).

At its best, software development is a procedure for optimally designing a product with interactive properties for supporting the performance of work tasks. Because computer science has accumulated quite a treasure trove of very broadly applicable algorithms, software development is increasingly focusing attention on those facets of application oriented software which are not amenable to algorithmic treatment. While the purely technical aspects of a software product are best dealt with by optimisation procedures attuned to a technical context, the non-technical context of the application environment aimed at requires the implementation of optimisation procedures of a different nature: moderation of human activities (Wohlgemuth 1995).

It would be false indeed to expect that at the outset of a larger reorganisation of a work system any single group of persons could have a complete, exact and comprehensive view of the ideal for the work system to be set up. Only during the analysis, evaluation and planning processes are the people involvable to develop an increasingly clear picture of what it is that they are really striving for. This is basically why the requirements of the applier seem to 'change' they do not really change but simply become concrete within the anticipated boundary constraints. This process of crystallisation should be allowed to unfold as completely, as pertinently and from a global perspectives inexpensively as possible. Completeness can be reached by ensuring that each affected group is involved at least through representatives. Iterative, interactive progress makes the ideal concept increasingly concrete (for more details see Rauterberg, Strohm and Kirsch 1995).

3. First steps to Implement Technology

The analysis phase is frequently the one most neglected (Boehm 1984). This is essentially due to the fact that methods and techniques need to be used primarily the way occupational and organisational sciences have developed and applied them (e.g., Macaulay et al. 1990). Inordinately high costs incur from the troubleshooting required because the analysis was less than optimal (Rauterberg, Strohm and Kirsch 1995). The time has come to engage occupational and organisational consultants at the analysis stage who have been especially trained in moderation techniques for software development! Once the analysis of the work system to be optimised has been completed, the next stage is to mould the results obtained into implementable form. Methods of specification with high communicative value are recommended here (Kraut and Streeter 1995), and not a method that enable the project management to retrace a misspecification (the 'traceability problem' see Pohl 1993). Sufficient empirical evidence has accumulated by now to show that task and user oriented procedures in software development not only bring noticeable savings in costs (Rauterberg, Strohm and Kirsch 1995), but also significantly improve the software produced (van Swede and van Vliet 1994).

How then, can the barriers mentioned above be overcome? The first thing is to determine 'if' and 'where' it makes sense to employ technology. "Although the view is still widely held that it is possible to use technology to eliminate the deficiencies of an organisation without questioning the structures of the organisation as a whole, the conclusion is nevertheless usually a false one" (Klotz 1991). The intended division of functions between man and machine is decided during the specification of the tool interface (for a more detailed discussion see Ulich et al. 1991 and Grote et al. 1995). Once those concerned are sufficiently clear about which functions are amenable to automation, the next step which should be taken is to test the screen layout on the end users with handdrawn sketches (the extremely inexpensive 'pen and paper' method, see Ehn and Kyng 1991).

The use of prototypes, to illustrate the dynamic and interactive aspects of the tools being developed, is indispensable for specifying the dialogue interface. However, prototypes should only be used very purposefully and selectively to clarify special aspects of the specification not indiscriminately. Otherwise there looms the inescapable danger of investing too much in the production and maintenance of 'display goods'.

A very efficient and inexpensive variation is provided by simulation studies, for example, with the use of hand prepared transparencies, cards, etc. which appear before the user in response to the action taken (Schuler and Namioka 1993). Simple and fast moderation techniques for involving users include discussion groups with various communication aids (Metaplan, layout sketches, 'screendumps', scenarios, etc.), questionnaires for determining the attitudes, opinions and requirements of the users, the 'walkthrough' technique for systematically clarifying all possible work steps, as well as targeted interviews aimed at a concrete analysis of the work environment (Macaulay et al. 1990, Nielsen 1993). Very sound simulation methods (e.g. scenarios, 'Wizard of Oz' studies, mockups) are available for developing completely new systems without requiring any special hardware or software (Ehn and Kyng 1991). Nielsen (1993) presents a summary of techniques for the analysis and evaluation of interactive computer systems.

4. Conclusion

One of the principal problems of traditional software development lies in the fact that those who have been primarily involved in software development to date have not been willing to recognise that software development is, in most cases, mainly a question of occupational and/or organisational planning and changing. A successful change of work and organisational processes and structures needs primarily the agreement among all involved groups. To come up with a shared understanding, the quality and the quantity of communication are critical success factors.

The moderation instead of modelling of human activities is of crucial importance. To change an organisation means to persuade people to behave in a new, different way than before. A model per se does not change anything; only a shared understanding among all affected parties is the basis of a real change. Participatory design and the adequate moderation of all necessary learning processes are the crucial success factors. Where information system development is approached from such a perspective, it would be planned from the beginning to engage experts in occupational and organisational planning in the project to moderate and co-ordinate the whole implementation process (see Kraut and Streeter 1995). This would require interdisciplinary cooperation between occupational and organisational experts on the one hand and software development experts on the other. The extensive qualification required in each of these fields makes it virtually impossible to dispense with such interdisciplinary cooperation. We must start learning to jointly plan technology, organisation and the application of human qualification.

References

Boehm, B., 1984, Verifying and validating software requirements and design specifications. IEEE Software 1(1): 75-88.

Booch, G., 1994, Object-oriented analysis and design (Menlo Park: Benjamin/Cummings).

Brunet, J., Cauvet, C., Meddahi, D. and Semmak, F., 1994, Object-oriented analysis in practice. In Advanced information systems engineering by G. Wijers, S. Brinkkemper and T. Wasserman (eds.) (Lecture Notes in Computer Science 811; Berlin: Springer), pp. 293-308.

Champy, J., 1995, Reengineering management: the mandate for a new leadership (New York: Harper & Row).

DeMarco, T. and Lister, T., 1987, Peopleware: productive projects and teams (New York: Dorset).

Ehn, P. and Kyng, M., 1991, Cardboard computers: mocking it up or hands-on the future. In Design at Work: Cooperative Design of Computer Systems by J. Greenbaum & M. Kyng (eds.) (Hillsdale: Lawrence Erlbaum), pp. 169-195.

Grote, G., Weik, S., Wufler, T. and Z–lch, M., 1995, Criteria for the complementary allocation of functions in automated work systems and their use in simultaneous engineering projects. International Journal of Industrial Ergonomics 16(46): 367-382.

Hammer, M. and Champy, J., 1993, Reengineering the corporation : a manifesto for business revolution (New York: Harper Business).

Klotz, U., 1991, Die zweite era der Informationstechnik. Harvard Manager 13(2): 101112.

Kraut, E. and Streeter, L., 1995, Coordination in software development. Communications of the ACM 38(3): 6981.

Macaulay, L., Fowler, C., Kirby, M. and Hutt, A., 1990, USTM: a new approach to requirements specification. Interacting with Computers 2(1): 92-118.

Naur, P., 1985, Programming as Theory Building. Microprocessing and Mircoprogramming 15: 253-261

Nielsen, J., 1993, Usability engineering. London: Academic Press.

Pohl, K., 1993, The three dimensions of requirements engineering. In Advanced information systems engineering by C. Rolland, F. Bodart and C. Cauvet (eds.) (Lecture Notes in Computer Science 685; Berlin: Springer),pp. 275-292.

Rauterberg, M., Strohm, O. and Kirsch, C., 1995, Benefits of user-oriented software development based on an iterative cyclic process model for simultaneous engineering. International Journal of Industrial Ergonomics 16(46): 391-410.

Schuler, D. and Namioka, A., 1993 (eds.), Participatory design: principles and practices (Hillsdale: Erlbaum).

Ulich, E., Rauterberg, M., Moll, T., Greutmann, T. and Strohm, O., 1991, Task orientation and user-oriented dialogue design. International Journal of Human Computer Interaction 3(2): 117-144

van Swede, V. and van Vliet, H., 1994, Consistent development: results of a first study on the relation between project scenario and success. In Advanced information systems engineering by G. Wijers, S. Brinkkemper and T. Wasserman (eds.) (Lecture Notes in Computer Science 811; Berlin: Springer), pp. 80-93.

Wohlgemuth, A.C., 1995 (ed.), Moderation in Organisationen: Problemse methode fur Fuhrungsleute und Berater (Bern: Haupt).

Womack, J.P., 1995, Neues von Hammer und Champy. Harvard Business Manager 18(1): 15-17.

Section II

PROBLEMS AND SOLUTIONS

Problems of incorporating Usability into the System Development Lifecycle

Martin C. Maguire
m.c.maguire@lboro.ac.uk
HUSAT Research Institute
The Elms, Elms Grove, Loughborough, Leics. LE11 1RG, UK

This paper discusses a number of problem areas when applying usability activities in the system design process or lifecycle. The paper is divided into sections relating to the main stages of design. For each section, suggestions are offered for enhancing the effectiveness of usability activities within design.

1. Introduction

Software quality is now defined in terms of six broad categories: functionality, reliability, efficiency, maintainability, portability and usability (Bevan and Azuma, 1997). Yet the process of incorporating usability activities within the development cycle is not straightforward. This paper discusses some of the problems that Human Factors professionals may face in trying to achieve usable systems. It draws upon HUSAT's experience of working with UK industry, and in supporting projects within the European Telematics Applications programme via the projects INUSE (Information Engineering Support Centres) and RESPECT (Requirements Engineering and Specification in Telematics). For further information see the INUSE web site at http://npl.co.uk/inuse

2. System concept

System development usually starts with an idea or concept, either to create or enhance an existing system. Human Factors specialists may be called upon to elicit reactions to the system concept from potential users via surveys or discussion groups. While this is a good approach, problems can arise with users not being able to visualise the new system or appreciate how it might operate, thus preventing them from commenting upon it effectively. For example, in one commercial study, carried out by HUSAT, users expressed considerable doubts about the safety of iris scanning technology for customer identification when using bank machines. Such concern was reduced when it was understood that the system used simple photography rather than a laser beam! Users could then concentrate upon other issues such as how bank customers would register with the system, what happens if the camera system breaks down, and what safeguards are needed to protect iris scan information that the bank might hold. Human Factors personnel can use their skills to help demonstrate the system concept, and explore the human aspects of its operation and how acceptable it would be in a real situation.

For larger bespoke systems, there is the danger that analysts will look at how the whole of current working system can be computerised rather than just appropriate parts. This can remove much of the flexibility of, say, the existing manual system. Again, by being able to communicate with future users, Human Factors personnel can help define the boundary between the social and technical components to create an efficient and usable whole system.

3. Requirements specification

Human Factors personnel are mainly concerned with user requirements within the system specification, e.g. functions needed to support the user task goals, general requirements for the user interface, user support needs, workstation, workplace and environmental requirements etc. Human Factors specialists will normally begin the user requirements specification by performing an analysis to identify all groups of users and stakeholders for the system. Although all groups may be recognised, a small group of people only may to be selected to act as user representatives for all groups. While the group will have a good understanding of their own areas of work, their appreciation of other stakeholders may be limited. Within stakeholder meetings, strong personalities may dominate, or junior members of staff may not wish to voice opinions when senior staff are present. Human Factors personnel may thus need to hold individual interviews with user and stakeholder groups to obtain a broad range of views.

For some companies, accessing end-users is difficult. This is particularly true when developing generic products which may be used by a range of people in different circumstances (Maguire and Graham, 1998). Experience with a UK telecommunications company has shown that development engineers have difficulty gaining access to end users of their products who may be employees within other companies. For generic products therefore, it may be useful to propose the creation of panels of representative end users drawn from members of the public, or recruited through employment agencies.

Damodaran (1996) highlights other issues. As the design develops, there is the danger that user representatives relate more with the technical design team than with the users, and become out of touch with working practices and problems. This is known as the 'hostage' phenomenon. User representatives may need to elicit information from other users and stakeholders but may fail without management support. Damodaran gives advice on the selection of representatives, obtaining support from other users, and the need for user representative training to learn about the various issues that will be under consideration at different stages of the lifecycle.

Another problem is managing user expectations. If the requirements process involves asking users to specify a wish list of what they would like from the system, it is likely that they may be disappointed with the result. It is important therefore to understand the design constraints and establish user requirements at an achievable level so that future users are not disappointed with the outcome.

User requirements may be recorded in the form of a large document, written in technical terms, which end-users are unable to understand. Users may then be required to sign off the requirements without fully understanding them. The Human Factors representative may need to act as the link between the designers and users, and to clarify the contents of the user requirements with diagrams, pictures, flow charts etc. Such informal representations can be very useful and give the end-users confidence that what has been specified is what they require.

4. Design and implementation

The use of rapid prototyping tools allows iterative design and the potential to create usable systems as part of an iterative design cycle. However when each new version of the prototyope is produced, it is important that the usability aspects are considered as of equal importance to the fixing of bugs. There is also the danger that users are allowed to comment on the prototype but are not allowed to interact with it. Such agreement by demonstration is rarely adequate for system development.

There are many sets of published guidelines on user interface design to support the design process. When Smith and Mosier (1984) surveyed of the use of interface guidelines among designers, they found that while 98% of respondents thought that design guidelines were needed, 63% reported that no guidelines were used. Reported problems of guideline usage were that they were often too general, too technology specific, or not relevant to the designer's application. Access to large documents containing guidelines also discourages use. HUSAT's own feedback from developers reveals that for disparate systems, it is hard to carry guidelines over from one system to another, so it is desirable to create specialised sets of guidelines for particular application areas. It has also been proposed that computerised tools incorporating interface guidelines would be an effective means of including them in the design process.

5. User testing

Testing a prototype version of the system with users is a common role for Human Factors specialists. A study by Dillon et al. (1993) of people involved in systems design found that the main problems in performing usability tests of system prototypes were: limited time and resources, limited Human Factors skills, knowing what aspect to test, lack of both usability metrics and a sound methodology. Methods and metrics have been developed for user performance (Bevan and Macleod, 1994) and subjective assessment (Kirakowski, 1996) which formalise usability measurement and offer clear metrics. However training and adequate resources are needed in order that these methods be used properly and to promote user evaluation skills generally. Experience shows that evaluation methods may also need to be adapted to particular situations. Discount usability methods (e.g. Nielsen, 1993) can also minimise the required time and resources for diagnostic testing. It is also important to know what is the scope for change so that user testing can focus upon appropriate issues.

6. Installation and maintenance

During installation and maintenance, Human Factors personnel can play a useful role supporting management of change. This involves a process of raising user awareness about the forthcoming system, establishing training requirements, and establishing mechanisms for user audit so that feedback is obtained when the system is installed. Problems and comments can then be recorded to be addressed within updates to the system.

7. Management commitment and training

Experience has shown that the commitment of project management is perhaps the main prerequisite for effective Human Factors involvement in systems design. This will allow time to be made available to perform usability related activities properly, and for user representatives to make an appropriate input. There is also the need to demonstrate the potential risk of not taking account of Human Factors in the lifecycle. An important way of transferring Human Factors knowledge into practice is to train members of the design team in methods of usability design so that they can practice Human Factors themselves. Experience has shown that companies are enthusiastic about taking up training but one-off introductory training courses are not necessarily sufficient. It is helpful to provide graduated training so that developers can specialise in particular areas (e.g. interface design, requirements specification etc.). Follow-up courses and clinics and workshops, where designers can bring and discuss current problems, provides useful reinforcement to training.

References

Bevan, N. and Macleod, M. (1994) Usability measurement in context, Behaviour and Information Technology, 13, 1 and 2, 132-145.

Bevan, N. and Azuma, M. (1997) Quality in Use: Incorporating Human factors into the Software Engineering Lifecycle, In: Proceedings of the Third IEEE International Software Engineering Standards Symposium and Forum (ISESS'97), 169-179.

Damodaran, L. (1996) User involvement in the systems design process - a practical guide for users, Behaviour and Information Technology, 15, 6 and 2, 363-377.

Dillon, A., Sweeney, M., and Maguire, M. (1993) A survey of usability engineering within the European IT industry - Current needs and practices, Proceedings of the HCI'93 Conference, J. Alty, D. Diaper and S. Guest (Eds), Cambridge University Press, People and Computers VIII, Loughborough, 81-94, September 1993.

Kirakowski, J., (1996) The software usability measurement inventory: background and usage, In: P. Jordan, B. Thomas, B. Weerdmeester and I. McClelland, Usability evaluation in industry, Chapter 19, 169-177, London: Taylor and Francis, ISBN 0 7484 0460 0.

Maguire, M. and Graham, R. (1998) A survey usability practices and needs in Europe, Contemporary Ergonomics, Proceedings of the Annual Ergonomics Society Conference, Cirencester, 269-273, 1st - 3rd April, 1998.

Nielsen J. (1993) *Usability Engineering*, London: Academic Press (AP Professional), ISBN 0-12-518405-0.

Smith, S. L. and Mosier, J. N. The user interface to computer-based information systems: a survey of current software design practice, Behaviour and Information Technology, 3,3, 195-203, 1984.

Communication between User and IS Analyst: Is Information Communication?

Marian Kuraś *eikuras@cyf-kr.edu.pl*
Agnieszka Zając *eizajac@cyf-kr.edu.pl*
Cracow Academy of Economics, Dept. of Information Systems;
27, Rakowicka Str., PL-31-510 Cracow, Poland

We cannot solve the problem
with the same thinking we had when we created the problem.
Albert Einstein

Introduction

Communication is the only way to learn **why** to design, **what** and **how** to design. This seems to be evident but still too often information technology (IT) does not meet needs of its users and does not give satisfactory outcomes. The role of communication in information systems (IS) development is discussed among the stakeholders in the process in specific circumstances of the cultural transformation. Poor communication is the reason for slow management changes, which is even more serious. The process of learning is spontaneous and usually disorderly[1] because of management insufficient effectiveness. Communication seems for that reasons to be a critical issue of management.
The paper presents common practices in IS design and usage in transforming economies. In the first section the background is presented. Next two sections discuss needs of users systems analysts. Sections 4 and 5 explain why both groups of process participants do not cooperate effectively and what should be done. The paper closes with conclusion with respect to improving communication among IS development.

Background

Communication is a 'new field' just disclosed to business people and specialists in different domains. Everything was well structured and easier in centrally planned economy. As changes began in 1990 things turned out to be much more complex and complicated. Previously, activities were totally regulated in detail and people were expected to behave according to the will of 'big brother'. Organizational communication was formalized and opinion that computers would replace employees was evident for many technologists. Even some researchers developed complex models of an automated organization. IT people and many managers accepted such visions of total computerization. It appeared to be adequate to their ideal world of algorithms and machines without erroneous and unreliable people (users).
This ideal world was destroyed at the beginning of 90'ies when central and east European nations took the way of democracy and market economy. Standardized organization, centrally planned economy and technology replacing people turned out to be biases. Technology has regained its supporting role. More managers/users begin to better understand their role and strive for learning how to apply IT and to get more of it. IT is becoming a powerful tool when used purposely by skilled people. In the other case it is just a very expensive gadget.

[1] People try to improve their communication skills reading for example '*10 minute perfect communication guide*' '*11 ways for communicate perfectly at work and in family and become happy forever*'.

What does a user need?
It is easy to answer this question: *a user wants a complex solution of an unspecified problem.* It should be inexpensive, easy to learn and to use, flexible and designed in very short by specialists who should know what to do, and do not ask millions of stupid questions. Users mostly do not try to communicate because they usually think they do not know enough about IT. In a fact usually they do not know what they need and cannot imagine how IT may support them in their work. These are fundamental barriers that make it impossible users to participate in IS development.

Users expect IT/IS professionals to take initiative and to suggest what are the problems. They do not understand IT so they cannot know how it may help to solve the problems. IT people do not usually help users to comprehend the role of IT as a tool, a new organizational technology. Users are afraid not only of IT but generally of anticipated complex change. It compounds power and status shifts, need to upgrade qualifications or possibly to have assignments changed. Usually users do not know what are the goals of the new IS and as well as what impact it will possibly have on their jobs. Managers in general have no vision of changes necessary to efficiently apply IT. They do not tell users about the change and neither do IS analysts. This limits communication and restrain organizational learning. The result is that poorly designed systems do not meet organizational needs.

What does an IS analyst need?
Also IS analysts seem to need from users nothing but a solution. It would be the best if this solution were represented in formal way. The problem is that the reason for users to address IS analysts were that they did not know how to cope in messy situations. Often they do not even know what are but they know the solution – computerization. IS people usually ask few questions about what users need. Managers had their idea what and how to change. Generally they want to minimize real changes and obtain effects in the way of replacing people by computers.

IS designers are results oriented. They want users to give them *all the necessary information* to complete solution and get users' acceptance. IS people give up requirements analysis and far too often they just describe how work is done. They then transform data structures and procedures, optimize them and finally present such new solution to a user. They do not solve the problem since in a fact only symptoms were described and inspected. The result is that problems persist, users blame IS developers for poor performance, IS designers introduce changes to software and argue that software is accurate to what users expected and finally accepted.

IS system developers are not interested in mission, business goals and objectives and strategy of an organization. They cannot though understand need for organizational change and the IT role as a change. In the result IT people produce a technical solution usually poorly meeting requirements. Additionally such solutions need numerous adjustments to the users' needs shortly after tool is in operation. Usually users of such software tools find them incomplete and imperfect (*still under construction*), and often very difficult to operate.

The common mistake of IS developers are four assumptions:

- we and the users know what is necessary to develop a new, satisfactory solution;

- we know what information we need and users are capable to deliver accurate information;
- users understand need for change, are aware of IT capabilities and comprehend what we need from them;
- information delivered is accurate basis for IS development and imprecise and lacking information is an excuse for developers in case the solution is faulty.

IS analysts are expected to guide users along the process of IS development. Such assumptions constrain communication and disable learning.

Why they cannot cooperate effectively?

System analysis is a learning process. Learning enables users and IS developers to acquire essential knowledge about organization: its goals, strategy and other components of an organization, business and information processes and information requirements. Knowing business, its goals and objectives, and strategy is indispensable to determine goals of an IS.

Learning effectiveness fully depends on communication abilities of stakeholders. They have to be aware of need for learning, trained, skillful and efficient communicators. It is agreed that participation of all the stakeholders in the process of development is essential, but it happens when there is intensive and purposeful communication.

Assumptions taken by system developers that information transfer from users pledges adequacy and sufficiency is misleading. The result is closed and defensive communication, transfer of limited and incomplete information.

Additionally, methodologies and techniques of requirements analysis are not broadly known and applied. This explains why IS developers reduce this stage of system development. Especially users do not learn methods and techniques. They are noticed as tools for IT/IS specialists.

When will users and IS people can become effective?

All the participants should finally understand that the objective of IT/IS applications is a radical improvement of business performance. This means that they should learn about IT as well as about business goals and strategy in order to bring about necessary changes. The change an organization needs is not only and primarily technological change but the strategic organizational change enabled by organizational technology.

Effective, open and intensive communication is the only way to identify the need for change Analysts should learn from managers & users how it not how it is but can be. This involves intensive, open, unlimited and purposeful communication among all the stakeholders. They all must communicate proficiently and develop communication skills. Collective trainings for managers, users and developers is the good way to improve process of IS development. Also students of computer science and information systems must learn communication to become professionals.

References

Kuraś, M. (1991). Communication in Design Team and with Users. Informatyka, No. 10 & 11.

Kuraś, M., Zając, A. (1998). Goals and Objectives of Teaching Communication to Management Students. Zeszyty Naukowe Akademii Ekonomicznej we Wrocławiu (in press) in Polish.

Nęcki, Z. (1996). Komunikacja międzyludzka. Kraków: Wydawnictwo PSB.

Sanderlin. R. (1982). Information Is Not Communication. Business Horizons (March-Apr.).

Use of Questionnaires in the Usability Testing of Public Automats

Sissel Guttormsen Schär
guttormsen@iha.bepr.ethz.ch
Institute for Hygiene and Applied Physiology,
Swiss Federal Institute of Technology,
CH-8092 Zürich, Switzerland

Introduction

Verbal responses to questions are the most widely used source of data in social research. Although questionnaire creation is itself a science, the employment of this knowledge in questionnaire design is often weak. Most readers know the feeling of being asked a question when the question does not really apply to you at all.

This paper aims at discussing the use of questionnaires for evaluation of software and hardware for public use. Ticket vending machines, automatic teller machines (ATM's) and information kiosks are typical products encountered by all kinds of people. It is obviously important that such products should be appealing and easy to use. The apparently easiest way to find out if a product complies to this is to ask its users.

The intention of objectivity is predominant in all kinds of testing, but is probably most difficult to achieve when using questionnaires as a test instrument. The problem about testing if a product is appealing and easy to use is that such requirements reflect individual qualities of the users as much as the product itself. Maybe the first basic error in applying questionnaires in this kind of usability testing is the assumption that one can test primarily the software in this way. The use of questionnaires in this context is then a compromise between asking an unstable medium (the users) to tell something about what they might not know (their perceptions and the product) and the benefit of using a test instrument which seems easy to produce, is easy to apply, and which has low costs.

Theory

Recognising the problems and their possible causes is the best precondition to conquer them. Therefore, a review of some basic principles concerning the methodology of questionnaire design is useful.

Knowing what you are testing, or what you intend to test, is the first and most important rule when using questionnaires. Theoretically this means that a questionnaire should be valid. *Validity* in popular terms is the extent to which you measure what you intend to measure. Among the different kinds of validity (e.g. internal validity, external validity, statistical conclusion validity) the construct validity is particular relevant for questionnaire design. The term 'construct' represents the idea we want to examine. Operational definitions of our construct include components that are not supposed to be included and exclude portions of the underlying construct that should be measured. Figure 1 illustrates this.

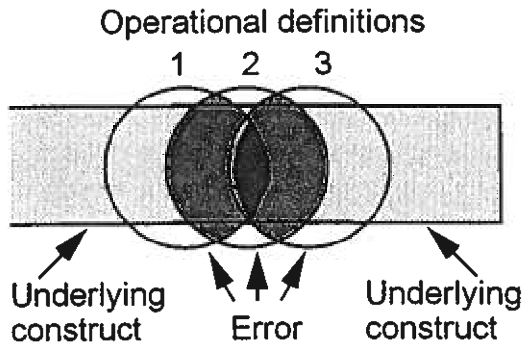


Figure 1: Construct validity

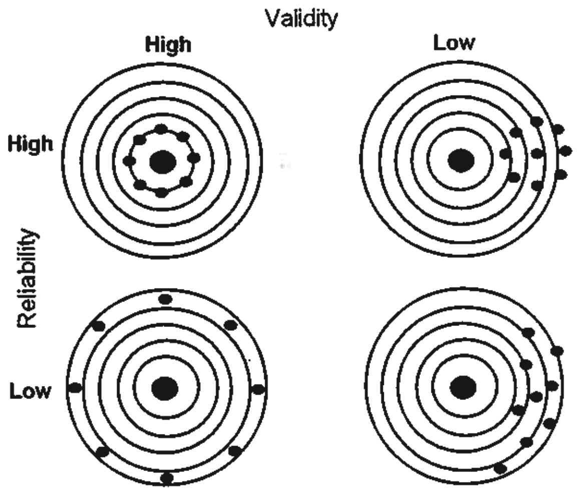


Figure 2: Different combinations of high and low validity and reliability

As the underlying construct cannot be tapped directly, but only indirectly through operational definitions, we can never be sure what portion of the construct the operational definition taps, and what portions are unmeasured. A classical example of an invalid test refers to the case of intelligence testing, where the subjects being tested must use a language that they do not speak perfectly. Then the intelligence quotient does not reflect the intelligence but an interference with language abilities. A *reliable* questionnaire shows the same profile between measurements when used under the same conditions. A questionnaire can be reliable but not valid. But, it is not possible to be not reliable but valid. The relationship between reliability and validity is illustrated in Figure 2.

In designing questionnaires for usability testing the recurring question should be whether the change or the effect we are measuring is due to features of the product we are testing or to people responding to something else. Are the questions we have designed capable of eliciting specific responses about the product we are testing?

Examples from field tests

Our experience is based on field studies from three different countries in Europe. In all studies questionnaires were an important source of data. The experience can be summarised as follows: asking people about their attitudes and understanding of computer applications for public use runs the danger of evoking answers motivated by other factors than the one intended. People responding to questionnaires have an unfortunate tendency to answer questions, even if they do not know the answer really. Further, when it comes to computers it seems that many people mean that they are less clever or intelligent if they find the product bad. They attribute low usability to themselves and not to the software. This can be illustrated by the examples shown in Figures 3 and 4. These examples are drawn from a field study of a public automat.

The questions are badly designed in many ways: first, they are formulated in a way that may give the user a feeling of insufficiency by negative responses. Second, it would be difficult to evaluate if the subjective impressions corresponded with the actual performance. Figure 4 illustrates peoples readiness to answer questions without knowing the answer really. The users spent very short time by the automat, and they were far away from trying out all its possibilities. Nevertheless, the majority regarded their short visit by the automat as having learned to use it.

I always knew how to cancel what I had previously selected.

Disagree strongly			Neutral			Agree strongly
1	2	3	4	5	6	7

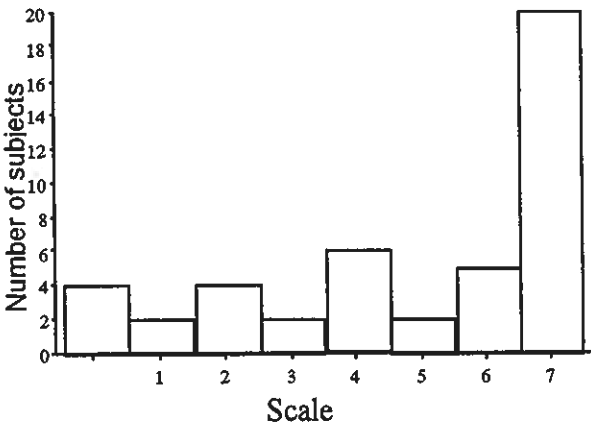


Figure 3: question and answers

I was able to quickly learn how to use the automat.

Disagree strongly			Neutral			Agree strongly
1	2	3	4	5	6	7

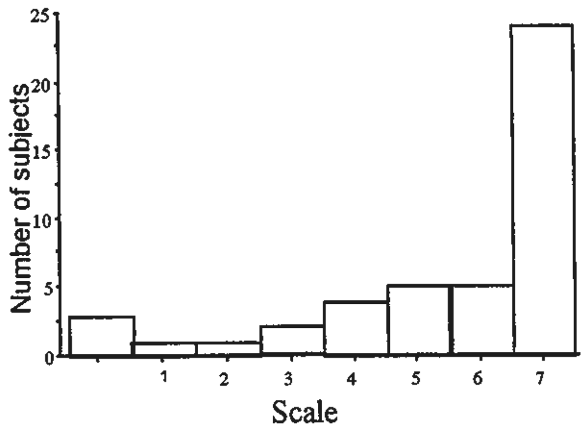


Figure 4: question and answers

Another phenomenon when using questionnaires for testing public automats is that the subjective experience from using the automat may be very different from the actual objective performance. In another study we tested the usability of a ticket vending machine with questionnaires and by recording log file data from the user actions. We found significant differences between the user groups with much and little computer experience. The first group reported that the time to get a ticket went quickly, and the latter that it took long time. This results did not correspond with the times recorded in the log files. There, no differences between these groups were found.

Discussion

Our experience from field studies demonstrates that it is important to take the validity problem serious. Further, it is necessary to have a clear opinion about what kind of data we want to record.

The validity problem is demonstrated with the two questions in Figures 3 and 4. What the users responded to in these cases, was probably not identical to the construct in the mind of those who designed the questions. Our experience from several usability tests in the field is that it is very difficult to design questionnaires that reflect qualities of the software or the product in focus. The problem is to formulate questions that can elicit the necessary information and at the same time being intelligible to all kinds of people. Because it is likely that the construct validity will be low, one should evaluate alternative ways of measuring. One alternative is to use log file data from the user actions as a reference for performance studies. In this instance usability can be measured by the ability to solve concrete tasks. In the lack of log file data different methods of observation and registration could be used (e.g. video, action registration matrixes). The drawback of this method is that the users do not act freely and self-initiated.

If questionnaires still must be used for collecting data about the effect of different features of the product, the design of the questionnaire should be based on knowledge of guidelines for questionnaire design and methodology. One way to overcome bad questionnaire design would be to use standardised and well proved questionnaires. The problem with standard questionnaires is, however, that they often are too general, or to long, going into details which are irrelevant for the actual product. Therefore, the design of a questionnaire for a special product is a recurring obligation. As a compromise we suggest that to use a standardised questionnaire (e.g. the user evaluation form in Schneiderman, 1992) and adapt this to the actual test situation.

The use of questionnaires for usability testing of public automats is good for the registration of subjective opinions and general acceptance, as well as for the registration of data about the users (age, sex, education, experience, etc.). A further powerful effect of questionnaires is that it can elicit responses which can differentiate the user group where objective data can not. This was the fact in our study where the experienced users reported longer time to get a ticket than the inexperienced user. This difference was unrelated to the objective time measures. A question is, if one can claim that subjective and objective factors are equal valid as a reference for user-friendliness. The subjective meaning about the product can be a more powerful determinant for user access than how the user actually handle it.

References

Holm,K. (1991). *Die Befragung 1*. Tübingen: A Francke Verlag GmbH

Kidder, L.M., Judd, C.M. (1986). *Research Methods in Social Relations.* New York: CBS Publishing Japan Ltd.

Rust, J. & Golombok, S. (1992). *Modern Psychometrics. The Science of Psychological Assessment.* New York: Routledge.

Schneiderman, B (1992). *Designing the User Interface. Strategies for Effective Human-Computer Interaction.* Massachusetts: Addison-Wesley Publishing Company.

Applying "House of Quality" to Software Development

Stanislaw Szejko
e-mail: *stasz@pg.gda.pl*
Department of Applied Informatics,
Technical University of Gdansk, Poland

1. Problem statement.

The paper aims at presenting a method for mapping user requirements to software quality characteristics. The problem has raised while the research aiming software quality evaluation and its related support system[1]. As defined in the project, a class of distributed management information systems (dMIS) is of the main interest.

Nearly all definitions of *software quality* relate it to a degree how software characteristics satisfy user needs [4, 8, ISO/IEC DEC9126]. Incorporating user requirements seems also crucial for assuring the quality of the initial phases, i.e. analysis and design (Fig.1)

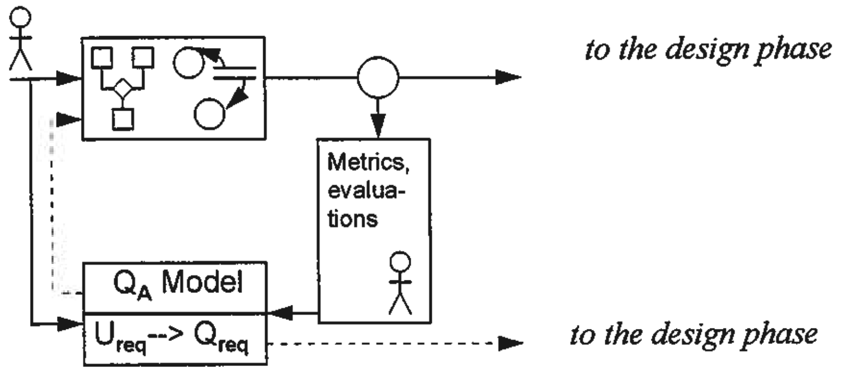


Figure 1. Quality evaluation model: situating the analysis phase model Q_A

The required quality evaluation model should combine five 'levels of the quality tree':

user requirements
quality factors (attributes)
quality criteria (characteristics)
quality metrics and/or evaluations
development phase products

[1] This research has been sponsored by the Polish State Committee for Scientific Research, grant No. 8 T11C 043 12.

2. A proposed solution.

Two classes of software quality models can be met typically [4, 6]:

the fixed approach models - they accept the fact that all the important quality factors that they need to monitor are a subset of those of one selected model;

the 'define your own quality model' approach, that accepts the general philosophy and decomposition, however the software designer and the user agree on which quality attributes they think are important for the given class of systems and a given product.

For the analysis and design phases we have decided on a kind of 'intermediate version' - for the assumed dMIS class of systems the model is going to be fixed, however it could be tuned (calibrated, weighted) with concrete user requirements.

The presented approach assumes adapting House of Quality (HoQ) matrix that was introduced by Quality Function Deployment method [5]. QFD consists of a raw of activities supported by its defined tables and matrices: House of Quality, Part Deployment, Process Planning, Production Planning. The approach helps to make the product developer sure about which of the quality characteristics or technical parts of the product should be increased starting with analysing customer requirements. The method was introduced for industrial and consumer products in Japan in 1972, next gained a lot of popularity in US. The approach has proven to be very successful in manufacturing and service environments, found a lot of applications, especially in car manufacturing and telecommunications, and it is believed to be equally beneficial for software products [2, 3]. QFD's House of Quality is used to determine user requirements, to prioritise them and to map the requirements to software characteristics thus calibrating the model according to currently specified user needs.

In HoQ customer requirements and product quality characteristics are linked in a table (Fig. 2). The left section of the table (denoted with '10' in Fig. 2) lists customer requirements and identifies strong, moderate and possible correlation with quality characteristics that are listed at the table vertical axis. The right section of the table ('1-9' in Fig. 2) lists on a scale of 1-5 customer requirements, company's current performance, competitors' performance, company plan and potential strong and moderate sales point aspects. It combines these weights into an absolute weight and a relative weight (a percentage). Result quality weights are collected by multiplying these correlated values by coefficients defined for mapping user requirements into quality characteristics ('11-12' in Fig.2).

The idea of adapting House of Quality table to software development assumes the requirements to be mapped to quality characteristics, thus influencing the attributes indirectly. A list of attributes and their characteristics has been derived from the works that were carried in parallel [1]; the main groups of user requirements for dMIS covered:

- ◊ functional requirements, grouping system tasks,
- ◊ non-functional requirements, like system security, performance, matching standards, etc.,
- ◊ cost (of development, of maintenance, etc.),
- ◊ development environment, including standards, technology, supporting tools.

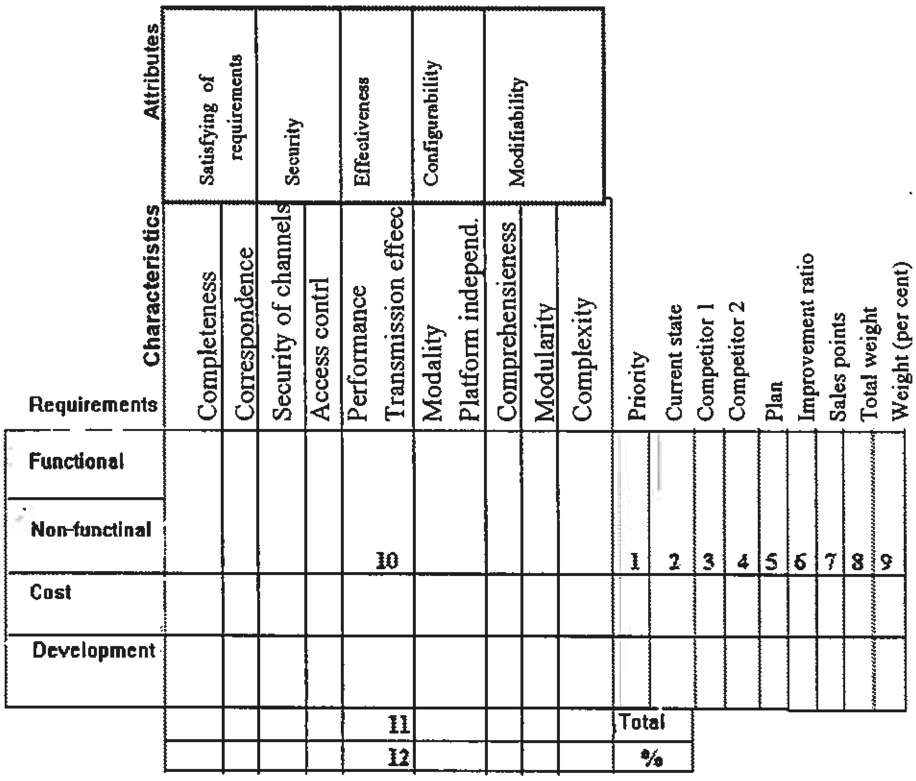


Figure 2. House of Quality defined to map user requirements to software quality characteristics.

The assumed idea of mapping the requirements to software characteristics allows for flexibility, thus the charac-teristics can be easily changed depending on the finally elaborated solution with no harm for the adaptation concept itself. As a matter of fact both, as the list of characteristics as requirements, are subjects to further investigation.

The approach has been verified by applying to an example of a payroll system for Gdańsk Post Office system [7]. Only functional, non-functional and cost requirements have been taken into consideration, a set of functional demands has been established according to actually expressed needs of the post office, non-functional demands for the class of distributed MIS systems were established and their correlation to software characteristics evaluated. Two other candidate systems were treated as competitors. The obtained weights for software quality characteristics are as follows:

Completeness	20	Clarity of documentation	2	Friendliness	7
Consistency and precision	5	Conformity	7	Modularity	2
Simplicity	5	Robustness	7	Readability	2
Traceability	1	Efficiency	4	Modifiability	5
Accuracy	8	Testability	2	Expandability	5
				Portability	10

The weights of the software characteristics obtained as the result of HoQ building process can be easily used for calibrating of the quality model.

3. Conclusion and future works

QFD's House of Quality seems to be strong enough mechanism to represent the influence of user requirements on software quality characteristics thus to calibrate the quality evaluation model for a concrete system with actual user needs. Provided the model attributes and characteristics are established we can try to establish the HoQ building procedure and prepare a related support system. A related prototype support system for building HOQ models of the QFD has been developed [7].

Bibliography

[1] Bobkowska A., Cysewski G., Szejko S.: SCSR grant No 8 T11C 04312 Reports No 72/97, 76/97, Gdansk, 1997

[2] Erikkson I. and McFadden F.: *Quality Function Deployment : a tool to improve software quality* Vol. 35. No. 9,1993, pp. 491-498.

[3] Erikkson I., McFadden F. and Tittanen A.M.: *Improving software development through Quality Function Deployment Proc.* In Wrycza S. & Zupancic J. (eds): V International Conf. on Information Systems Development, Gdańsk, Sept 1996.

[4] Fenton N.E.: *Software Metrics. A Rigorous Approach*. Chapman & Hall, 1993

[5] King B.: *Better Design in Half Time*, GOAL/QPC, 1989.

[6] Lukawski A.: *System SOA for Evaluation of Applications* (in Polish), MSc thesis, TU Gdansk, Department of Computer System Architecture, Gdansk, 1996

[7] Paul S.: *Graphical Support for Information System Evaluation Based on QFD Method*. MSc thesis, TU Gdansk, Department of Applied Informatics, Gdansk, 1997

[8] Pressman R.S.: *Software Engineering. A Practitioner's Approach*, McGraw - Hill, 1992

Usability Evaluation Methods for World Wide Web Sites

Nigel Claridge
nigel.claridge@nomos.se
Nomos Management AB, Sweden
http://www.nomos.se

Introduction

The explosion of World Wide Web Sites in general and from our perspective in Sweden has created a demand to assure the quality in use of the sites - do they work for their intended purpose. The interest in web usability has emerged from the more complex and interactive sites such as Internet banking, Internet employment services and sales and marketing sites to complex information provider sites. It has become clear that web sites must work quickly and efficiently for its intended target group. If they does not, a competitor site is but one click away.

At Nomos, we provide a range of design support and usability evaluation services to a wide range of web clients, both owners and producers. To support these services we have modified, adapted and extended a number of well used and practical methods and tools developed to improve the usability of traditional software. Most of our methods are adapted and extended versions of those we have applied for some time in 'traditional' software interface testing. Many of these tools and methods have been developed around International Standards concerning software ergonomics (for example, ISO 9241 Parts 10-17).

Our 'tool kit' also focuses on the user centred design structure defined in the international standard (draft) 13407 and our objective is not only to improve the web site itself but also the process by which the web site is created. This presentation focuses on the usability evaluation methods and the way in which they have been extended to suit the new development environment.

Specifically, we have worked (or are working) on large Web projects for Swedish banks, the Swedish Labour Market Board, several international investment companies, and a large international software corporation, amongst others. The majority of work has been with Internet Web sites, but with a recent growth in Intranet work.

We have experience in applying the following usability testing methods and tools to World Wide Web sites:

- ☐ Expert assessments/inspections, with reference to principles adapted and extended from those set out in ISO 9241 Part 10.
- ☐ Participatory evaluations (experts with users).
- ☐ Small and large scale usability lab tests with representative users carrying out representative tasks on web sites based on PMM (Performance Measurement Method).
- ☐ A specialised Web logging tool which captures a very rich set of information about how users interact with, and navigate through, a site.

- ☐ An online subjective usability assessment questionnaire - WAMMI - specifically developed for the Web, based on SUMI (Software Usability Measurement Inventory).

The methods and tools we employ for a given web site naturally depend on it's state in the development process and individual client requirements. In addition to testing methods, we employ context and requirements analyses as necessary, and a variety of prototyping techniques, as part of a user-centred Web design and evaluation process we use when working on development projects.

Expert assessments/inspections

These are the most frequent types of evaluation we carry out and are based on expert knowledge and principles adapted from ISO 9241 Part 10. They are used at intervals throughout the development process when we work with clients developing a new site. An expert evaluation is often combined with a small context study, the combined results of which give a good picture as to the usability of the site. This can then form the basis for planning future design activity.

Advantages:

- ☐ Relatively quick and inexpensive - the extent of the evaluation and the number of experts used (typically 2 or 3) can vary depending upon the nature of the site and client requirements.
- ☐ Can be carried out on paper or incomplete prototypes.
- ☐ Captures usability problems which might not be found during lab tests using a specific environment (e.g. issues concerning differences between browsers, accessibility for disabled users).
- ☐ Gives quick feedback into the development process - crucial in the short development time scales for Web projects.

Disadvantages:

- ☐ Requires a working technological knowledge of certain aspects of the web (for instance, unlike 'normal' software interfaces, the expert needs to be aware of differences in the way interfaces may appear on different browsers). This is especially important if useful recommendations are to be made as a result of the inspection.
- ☐ Whilst usability issues can be identified, in some cases it is difficult to judge how much trouble certain problems will cause users, such as aspects of navigation support.

Participatory evaluations (experts with users)

In these evaluations we evaluate a web site together with representative users. Usability experts will 'pair up' with representative users who will work together with the web site. This method is often employed with Intranet sites, or sites with specific applications, where representative users can easily be found, and a set of representative tasks can be pre-defined for users to work through in the evaluation.

Advantages:

- ☐ Involving representative users enriches the experts' feel and understanding of the way in which users use the Web site and the interaction issues concerned with it.

- ☐ Can be used as a method to work with users when only partially-working prototypes are available.
- ☐ Gives quick feedback into the development process - crucial in the short development time scales for Web projects.

Disadvantages:

- ☐ By having experts working with users, the users' way of working may well be affected.
- ☐ Users may become overly-critical of interface issues.

Small and large scale usability lab tests

We carry out usability 'lab' tests with users in the Nomos usability lab - a fully equipped usability lab designed originally for software testing. Typically, users have a period of 'free interaction' with the web site being tested, followed by a series of pre-defined representative user tasks which may include searching for particular information, working with particular applications, etc. Usability tests are combined with user interviews and WAMMI.

Advantages:

- ☐ User interaction with a web site can be observed in detail.
- ☐ Interaction is recorded and can be analysed later.
- ☐ Performance, efficiency and effectiveness measures can be made for particular tasks.
- ☐ "Lab tests" have a perception of high validity on the part of clients.
- ☐ Video recordings of usability problems can be very persuasive in convincing developers of the need to make changes to poor interfaces.

Disadvantages:

- ☐ Relatively time consuming (and thus costly).
- ☐ Requires recording equipment.
- ☐ Difficult to use with prototype sites and therefore usually used only towards the end of the development process, or on existing sites.
- ☐ User interaction may not be completely natural.
- ☐ May not capture usability problems experienced by users using a different Web browser or computer platform to that used in the test (less of a problem for Intranet sites, where such factors are known and the test environment constructed accordingly).

Specialised web interaction logging tool

In partnership with a large computer science research institute in Stockholm, Nomos is developing a logging tool which enables detailed logging of user interaction with a web site which is much richer and more detailed than standard server log files. The logging tool and analysis software has been specifically developed as a means of moving the software usability test laboratory on to the Web itself.

Advantages:

- ☐ Useful in analysing the way in which users navigate around a web site.
- ☐ Only requires the user to use the web site as normal.
- ☐ Data collection is automatic once specified and thus data from a large number of users can be obtained relatively inexpensively.

- ☐ Can be combined with results from online subjective assessment questionnaires to give a rich picture of web site usability.

Disadvantages:

- ☐ The logging may have some affect on performance of the Web interface
- ☐ Can produce vast quantities of data and thus requires care and some expert understanding in the specification of logging parameters and subsequent analysis.
- ☐ Requires a finished prototype or existing site.
- ☐ There are limits to the logging abilities when using Java applications.

Online subjective usability assessment questionnaires

In partnership with the Human Factors Research Group in Cork, Ireland, Nomos is developing a generic web-administered user satisfaction questionnaire for web sites - WAMMI - which will give statistical measures of software usability on five scales - "User efficiency", "Affect", "Helpfulness", "Control" and "Learnability", as well as a global measure. WAMMI will, after a series of trials in Sweden, Norway, Australia, Holland and the UK, be commercially available after the summer of 1998.

Advantages

- Easy to administer on the web.
- Reflects real user experience.
- Can be used as a basis for comparing web sites.
- Gives quick feedback.

Disadvantages

- Can be difficult to get users to answer the questionnaire.
- Doesn't identify specific problems, although careful analysis may point to them.

☐

Assessing Satisfaction with Consumer Electronics

Gerard Hollemans
hollemans@ipo.tue.nl
IPO Center for Research on User-System Interaction
PO Box 513, 5600MB Eindhoven The Netherlands

Abstract: An attempt to develop a tool for assessing the satisfaction with a consumer electronics product is presented. First the need for such a tool is outlined, followed by a description of the requirements for the tool, and a discussion about the suitability of available tools. Finally, the construction of an itempool for a satisfaction questionnaire and the refinement of this itempool are described all of which is followed by some concluding remarks.

The Need for a Usability Evaluation Tool

Usability is no longer a relatively unimportant add-on to a product. As products in the domain of consumer electronics no longer differ substantially in functionality, user-friendliness has become increasingly more important [4]. Consequently, there is a need to assess the usability of the products that are being developed. This need was also felt by a large manufacturer of consumer electronics in The Netherlands. In cooperation with IPO Center for Research on User-System Interaction a tool to assess the usability of consumer electronics products is currently being developed. In this paper the ongoing development is described.

Requirements for the Tool

The products under consideration are from the domain of consumer electronics. They are used by non-professional users for pleasure in a 'home situation'. Because the usage of the products is discretionary, subjective usability measures are preferred [1]. Although there are many techniques to obtain subjective measures, the selected measurement technique is the questionnaire. A well-designed questionnaire can be administered relatively fast and easy and this holds for the data analysis as well. Furthermore, a questionnaire requires relatively little training to administer properly. Since there is usually little time and a small budget to perform a usability study, these characteristics are quite convenient.

Apart from the practical requirements, there are also theoretical requirements. The data resulting from the evaluations have to be reliable and valid, based on some theory, and statistically analyzable. In addition, the questionnaire has to be formative, i.e. it has to deliver information that helps to improve the product that is tested.

Currently Available Usability Questionnaires

Although a number of usability questionnaires are available today, there is no questionnaire that fulfills all requirements stated above. The main problem is the type of product that is tested. Many of the questionnaires concern the usability of software. This poses a problem for two reasons: The validated questionnaires are validated for software products and there is no guarantee that the validity will hold for consumer electronics as well. Furthermore, some of the items are not suitable for consumer electronics. For instance, items about menus, data files and help functionality do not apply to CD-players or tuners. Therefore, questionnaires that measure the usability of software can not be used without modification.

The remaining questionnaires, which are not focused on software, have other shortcomings [2] and therefore a way to cope with the absence of a perfectly suitable questionnaire had to be found. There are two options: adapt one of the available questionnaires or construct a completely new questionnaire. Adaptation of one of the available questionnaires is not very efficient because it essentially means constructing a 'new' questionnaire after all. In classical testing theory, which is used for the currently available questionnaires, the reliability and the validity of a questionnaire change when the questionnaire itself changes [3]. Since modifying a questionnaire is essentially equal to constructing a new questionnaire, it was deemed best to construct a new questionnaire that complies to all the requirements completely from the start.

The Itempool

To base the questionnaire on the knowledge and opinions of the experts of the cooperating company and to get some commitment to the questionnaire, their experts were involved in the definitions of the dimensions of usability. This was done by a small scale survey among the experts. In this survey the experts were asked to indicate to which aspects of the user-system interaction (focuses) the listed qualities (dimensions) apply. The dimensions were defined based on these data. Later this survey was extended to include other experts by way of a mailing list on the internet. Their answers did not differ substantially from the answers given by the experts of the cooperating company. A more detailed description of the experiment can be found in a report by DeRuyter and Hollemans [2].

The dimensions and focuses were derived from the literature. From a number of sources that discuss aspects of usability dimension names and, if available, descriptions of those dimensions were extracted. On that basis a set of dimensions was drawn up to cover all dimensions that were implicitly or explicitly mentioned. Along with the set of dimensions a set of focuses was created, so that all descriptions of the dimensions that were extracted from the literature could be characterized by one or more combinations of a dimension name and a focus. For instance, the description of Learnability might be, that 'the layout of a product must enable the user to learn to perform tasks with the product quickly and with minimum effort'. The dimension then is Learnability and the focuses are Layout, Task, and Effort.

Items were phrased for nearly all dimensions, 134 items in total. Some dimensions were taken together for practical reasons: The items for Transparency and Self-descriptiveness as well as the items for Reliability and Robustness were considered to be too identical to warrant separate dimensions. Furthermore, the dimension Suitability was discarded. Suitability, according to the consulted experts, concerns the degree to which a product is apt for a task and complies to all sorts of requirements. However, items like 'The CD-player lets me completely perform entire work routines' or 'The video recorder is well suited to the requirements of my work' are not appropriate for consumer electronics as the purpose of these products is to entertain rather than perform. How well the product entertains is considered to be beyond the scope of Suitability.

First Results

The itempool had to be refined to separate the good items from the deficient items. This was done by means of an experiment in which the participants were asked to interact with two different CD-players by performing some tasks. After interaction with one of the CD-players the participants were asked to respond to all items by indicating their agreement with the item on a 5 point rating-scale. These responses as well as a direct rating of their appreciation of the product on a 5 point scale were recorded. This procedure was repeated for the other CD-player.

Currently the data are being analyzed and the intermediate results look very promising regarding reliability and validity. A Cronbach's alpha of about 0.95 demonstrates that the questionnaire has a high degree of internal consistency. In addition, the responses to the items correlate quite well with the direct rating of the appreciation of the products (the average multiple correlation coefficient is 0.80). These statistics refer to the 29 questions that remained in the itempool after screening all items on the comments that participants made on them. For some dimensions the remaining number of items was too low (for example 2 or 3 items) and therefore new items had to be phrased. As a result the itempool now consists of 48 items.
Until now, nothing has been said about the formativeness of the questionnaire. To make the questionnaire potentially formative a series of scores will be computed, one score for each dimension, and together these scores will form a profile. This profile of scores shows the relative strengths and weaknesses of the product. This is similar to what is done with the scores of the SUMI [5]. Whether this profile of scores indeed delivers information that helps to improve the product remains to be tested.

Concluding Remarks

Although the (statistical) results look promising, the theoretical basis of the questionnaire is a point of concern. There is no established theory concerning the determinants of users' satisfaction with consumer electronics. Consequently, there is no established principle to guide the selection of items. Here, a practical approach is taken by involving experts of the cooperating company and relying on personal judgments. Furthermore, a theory concerning the determinants of users' satisfaction with consumer electronics is required to allow for more formal testing of the validity of the questionnaire.

References

1. Bevan, N. and Macleod, M. (1994). Usability measurement in context. *Behaviour & Information Technology 13(1/2)*, 132-145.
2. DeRuyter, B.E.R., & Hollemans, G (1998). *Towards a User Satisfaction Questionnaire for Consumer Electronics: Theoretical Basis* [NL-TN 406/97]. Eindhoven: Nat. Lab. Philips Electronics N.V.
3. Hambleton, R.K., Swaminathan, H., & Rogers, H.J. (1991). *Fundamentals of Item Response Theory.* Measurement Methods for the Social Sciences. London: SAGE Publications.
4. Jordan, P.W. (1998). Human factors for pleasure in product use. *Applied Ergonomics 29(1)*, 25-33.
5. Kirakowski, J. (1996). The software usability measurement inventory: background and usage. In P.W. Jordan, B. Thomas, B.A. Weerdmeester, & I.L. McClelland (Eds.), *Usability Evaluation in Industry* (pp. 169-177). London: Taylor & Francis Ltd.

Learnability of Consumer Electronics

Mili Docampo Rama
IPO: Centre for Research on User-System Interaction
PO Box 513, 5600 MB Eindhoven The Netherlands
docampo@ipo.tue.nl

Abstract: Many adults complain that domestic products manufactured today are more difficult to use than earlier products. Therefore, the purpose of the project described in this paper is to analyse the extent to which user background- and product-characteristics play a role in the usability of consumer products. The aim is to formulate guidelines for designing interfaces that correspond with the way generations of users learn to use a device. Different strategies are used to transfer the knowledge acquired to designers. These strategies are discussed in this paper.

Introduction

In the project "Philips Technology Generations", several partners, namely Philips Design (Human Behaviour Research Group), University of Utrecht (Department of Sociology), Eindhoven University of Technology (Department of History of Technology and Gerontechnology) and IPO (User Centred Design Group) investigate which factors play a role for older consumers in the acceptance or resistance to buy and use (new) consumer electronics products. The focus will be on the telephone, television and videorecorder. The context of buying and use, the user's preference concerning the look and feel of the products, and the impact of technological changes on the user's behaviour are the domains of analysis. IPO focuses on factors that play a role in the learnability of new consumer products. Therefore, this position paper will focus mainly on the latter part.

Overview of studies

Domestic products have been around for many years now. In these years, their appearance and functionality have changed a lot. Nowadays, many adults seem to have difficulties learning to use modern consumer products (Stewart, 1992). Which factors contribute to these learning difficulties? Learning problems may have arisen because of (1) the increasing complexity of interfaces over the past years, or (2) age-related cognitive changes, or (3) generation-related differences in experience with technology.

Until now, three studies have been operationalized. The objective of the first study was to investigate whether age, being a member of a technology generation[1], and interface style play a role when learning to use a consumer product (simulation of a videophone interface featuring a single-layer and a layered interface). This study was concluded and confirms the hypotheses mentioned above.

At the moment, two follow-up experiments are running to analyse in detail the impact of the user's cognitive abilities and experience with technology on learnability. One of these experiments makes use of (1) cognitive tests that measure the user's cognitive abilities, and (2) a questionnaire that measures the user's experience with technology. These aspects are related to the user's learning behaviour during the interaction with a simulated multi-layered mobile phone. For the second follow-up experiment a television interface with a menu-structure and a corresponding remote control is simulated. In this study the learning behaviour of subjects with certain experience with technology is compared with that of inexperienced subjects. Whether these background characteristics explain the age-group related learning difficulties remains to be analysed.

Strategies of transferring knowledge

The purpose of the project is to transfer information acquired during the studies to designers and marketing staff. An overview of the strategies used until now to transfer knowledge to designers and the outcome is presented here.

Designers

The type of guidelines that is useful for designers depends on the developmental stage of the product. Since our project is not linked directly to product development projects at Philips, more indirect strategies have to be used to achieve guidelines that are suitable and useful for designers in various phases of the Product Creation Process.

The main focus at the beginning of the project was to get designers interested in our project by (1) setting up meetings within Philips Design to raise awareness, and (2) asking designers to co-operate with us in the project.

The meetings were successful in raising awareness and interest. The second strategy has only been tried on limited scale and was not working in this stage of the project. Designers were very reluctant to co-operate. A reaction such as "It is very interesting, but come back for co-operation when you have some real results" was very typical. So at this point, the project is known within Philips, but there is no direct involvement in the Product Creation Process, yet.

[1] According to Sackman, Hηttner and Weyman (1993) a technology generation is a cohort of people that experienced the availability of the same types of domestic appliances during their youth. As a consequence they show a similar way of using products nowadays.

This seems mainly due to a lack of direct relevance and applicability of the information in the design process. The information we offered at this early stage was probably too general, while the designer's information need is often of a specific nature.

For communicating the initial results, two strategies have been followed so far: (1) reports were written about the purpose of the study, the outcomes and some general implications for design, and (2) personal talks with overhead slides were given.

Reports of research groups within and outside Philips are often written according to a standard template. Such reports contain about thirty pages in which the theory behind the problem definition is explained extensively. Besides, the experimental design and the analyses are described in these reports according to scientific rules, which implicates a very technical and detailed description of the study. Finally, the implications of the results for further research are only presented at the last page. Therefore, reports were distributed on a small scale and probably only read on a superficial level. The second strategy was more successful. Personal talks offer the opportunity to match mutual interests and problems. So, again there were enthusiastic reactions. Designers mentioned to be open for a possible co-operation in the near future.

At the end of this summer, most results of the studies in the overall project will be analysed. Therefore, we are preparing new strategies. The aim is to develop strategies that transfer information to designers in such a way that it is more directly relevant and applicable in their design projects. Additionally, these strategies should transfer information in a more durable way. Seminars and training sessions may be a solution to these problems.

Conclusion

Much effort has to be put forward when research groups are not directly related to designers and the Product Creation Process. In such a case, along with the development of the research project, different strategies have to be used to transfer the information to designers.

References

Sackmann, R., Hüttner, B., and Weymann, A. (1993). *Technisierung des Alltags: Generationen und Innovationen*. Bundesministerium für Forschung und Technology.

Stewart, T. (1992). Physical Interfaces or "obviously it's for the elderly, it's grey, boring and dull". In: H.Bouma and J.A.M. Graafmans (eds), *Gerontechnology*. Amsterdam: IOS Press.

Product-Oriented and Process-Oriented Usability Assurance

Marcin Sikorski
msik@zie.pg.gda.pl
Technical University of Gdansk, Poland
Faculty of Management and Economics, Ergonomics Dept.

Introduction [1]

In recent years usability engineering has developed a number of techniques and tools aimed to improve quality of use for software products. These techniques and tools are primarily oriented to provide designers with tools for better design of user interface, to evaluate usability of a product or to include user-centered design requirements into existing software development process.

However all available usability techniques are aimed to help in developing a product which better fits to users' needs, they vary much in a way how they could be applied - by whom and what outcome can be expected by system developers.

Available approaches to usability assurance can roughly divided in two groups:

- oriented on *quality control*, that means checking whether the given product satisfies expectations and requirements captured from users, documents, market etc; this approach will be called here *product-oriented*, because the focus is entirely on the features of the product, separated from manufacturing environment in which this product was created;
- oriented on *quality assurance*, that means approach oriented not on correcting but on preventing usability errors by appropriate management of design and development process, and by integrating user-centered actions into design process, as its integral and inseparable part; this approach will be called here *process-oriented*.

Product-oriented approaches

Product-oriented approaches focus on evaluation the product and on specifying recommendations for usability improvement. Following examples of these techniques can be given:

a) for supporting user interface design:

- design guidelines, based on human factors knowledge base and dialogue design principles, like Preece (1994) or Newman and Lamming (1995),
- style guides and other documents developed for guiding screen and user interface design, like Microsoft (1993) or IBM (1992),

[1] This research has been sponsored by the Polish State Committee for Scientific Research, grant No. 8 T11C 043 12.

- software tools for guiding user interface design (IDA - Reiterer, 1993), or user interface management systems;
- b) for <u>evaluating</u> software product:
- checklist-based techniques, like Ravden and Johnson (1989) or EVADIS (Oppermann and Murchner 1992),
- user-based testing for product acceptance and usability;
- expert-based evaluation of user interface design or whole product for:
  - conformance with guidelines, style guides or standards,
  - usability in task context;
- c) <u>certification</u> of software products:
- by eligible institutions, like TÜV, for compliance with ISO 9241-11,
- by commercial service laboratories,
- by computer magazines.

All product-oriented techniques and approaches compose a set of different methodologies, varying in applicability and performance, so they support design and evaluation only in selected areas. Considered usability characteristics are usually separated from technical parameters of the product. As a result, use of product-oriented techniques is not systematic and in practice depends only on willingness of project managers or developers, being also prone to different pressures (time-to-market, cost), often tending to push usability issues aside.

Process-oriented approaches

Process-oriented approaches in usability engineering always address software life cycle, which describes subsequent phases in which particular software product or interactive system is manufactured. The most popular software lifecycle models specify following phases: Requirements, Analysis, Design, Implementation, and Installation.

Process-oriented approaches attempt not to *control* product usability, but to install into the software development process such mechanisms, which *assure* that high usability will be in this process achieved in a natural way.

Integrating usability tools with software process can be illustrated by following examples:
a) building into the process <u>checkpoints</u> for systematic evaluation and testing of the product of each phase; this is performed by a series of inspections and reviews:
- technical reviews for software requirements,
- usability reviews for user requirements (users involved);

b) implementing systems for systematic <u>monitoring</u> quality and usability characteristics, based on a clear hierarchy: criteria/attributes/characteristics/metrics; such systems maintain a database of evaluation results and a reference models for product quality, specified for different group of users -- or other stakeholders of the project;
c) <u>user centered</u> approaches, like:
- participative design, which involves users for the whole project duration, from the very early concept, till the operation of the final version of the product,
- based on ISO 13407 -- or another concept – idea of building into the process *user feedback* as a manageable system for systematic collecting, analyzing and following user's opinions, comments and suggestions; different information sources can be used and also methods adapted from marketing or consumer research;

d) certification of software development process:
- according to ISO 9000: whether software development process satisfies quality assurance requirements, as described in ISO 9000-3 or DIN ISO/IEC/12119;
- according to CMM (Capability maturity Model, Herbsleb et al., 1997), which certifies degree to which particular software development process is able to deliver products of demanded quality characteristics.

Process-oriented approaches seem to attract more and more interest recently, but they also meet problems in implementation, what will be shortly discussed below.

Observed limitations

A number of barriers can be observed when trying to implement process-oriented approaches (problems with product-oriented approaches are widely known, so they will not be discussed).

First of all, project stakeholders like developers, managers, users, investors, consultants, represent very different perspectives as to expected outcome. Their attention is focused on specific functional preferences (Sikorski, 1997), like:
- software engineering (technical, technology-oriented),
- human factors (user-centered),
- managerial (time- or cost-focused).

Tab. 1. shows that different "languages" can be used for describing specific phases of design, with different scope or meaning of the same words, and even some missing areas. Software engineering terminology seems to be the most complete and precise, although this perspective has been often blamed for neglecting users' needs in the product concept.

Table 1. Terminology for describing phases of software lifecycle.

Human-Centered Design ISO 13407	*User-Centered Design*	*Usability Engineering lifecycle*	*Project Management lifecycle*	*Software Engineering lifecycle*
Analysis	Definition	Analysis	Feasibility	Requirements Specification
Design	Concept	Design	Analysis Design	Planning Design Verifying
Evaluate	Evaluation	-	Programming	Developing Testing
-	Application	Implementation	Implementing	Implementation Integration
-	-	Installation Maintenance	Support	Installation Operation Maintenance
-	-	-	-	Retirement

A number of organizational, managerial, cultural and technical problems can be expected when trying to build usability engineering tools and techniques into organizations and their software development processes. Some of the commonly experienced problems:

- usability techniques are unfamiliar for most of developers, they lack access to usability tools or lab, also do not have time for usability training unless absolutely needed,
- most of developers are afraid to face real users, record and process their comments; lack of usability culture and contacts with real users results in cutting off the developers from their market; they also have no previous experience with user testing or surveying;
- managers do not see usability as a potential to build a competitive advantage for their products, and efficient strategy for advertisement.

Usability consultant is often a person whose professional help can not only improve a single product, but can also make the whole manufacturing process more user-centered and cost-effective. Many experiences show, however, that potential of usability consulting in many areas has not been yet fully utilized and can be still improved.

When serving industrial customers, an important strategic decision must be made: whether our consulting will be product-oriented or process-oriented. Depending on client company perspective and needs, appropriate actions will be proposed. Product-oriented usability consulting seems to be an easy choice: this job is simpler, repeatable and usually supported with some experience and traditions. Oppositely, process-oriented usability consulting requires more effort and knowledge from the consultant, regarding also some expertise in programming technology, CASE tools and software project management. This shift of focus is difficult for many usability consultants, especially if they are not engineers by their educational background.

Developing efficient communication with project managers and developers seems to be crucial for success, and for the art of "selling" usability technology and its benefit into organizations, where people usually think of project success in terms of expected outcome and resources, time or costs involved. Usability consultant must often think and speak the same way as his client does and recognize similar priorities, if he/she wants to be accepted as a partner – and even stakeholder – in the project.

Opportunities

One of the most efficient strategies for selling usability into organizations seems to be incorporating usability issues into different quality assurance programs, adopted in software industry. Certification of manufacturing processes gives a chance to include usability as an important aspect of software product quality. User feedback is required by several documents as a component of manufacturing process, by ISO 13407 and CMM in particular.

Because most of project managers in software industry remain skeptic as to applicability of ISO 9000 for their domain, CMM model offers more opportunities for usability transfer, and deserves particular attention from the side of usability specialists. CMM model specifies five levels of maturity of software development processes; these levels describe organizational and manufacturing culture, resulting in likelihood that delivered product will meet users/customers requirements, regarding also usability and acceptance.

Tab. 2 (based on Herbsleb et al., 1997 and Bevan, 1997) shows relationship between CMM levels and types of "usability culture" characteristic for software organizations.

Tab. 2 . Relationships between CMM and Usability Maturity models

CMM level	*Focus*	*Process*	*Usability*
1. Initial	Competent people and heroics	Ad hoc, intuitional, often chaotic, success dependent on a chance	Usability is not performed and ignored
2. Repeatable	Project management processes	Tracking costs, schedule, functionality; trying to repeat success of previous projects	Usability is common sense
3. Defined	Engineering and organizational support	All engineering and management activities are documented; approved standards only in use	Usability unfocussed, sporadic user involvement
4. Managed	Product and process quality	Detailed quantitative measures of software process and product quality are collected	Usability integrated, systematic and measurable
5. Optimizing	Continuous process improvement	Quantitative feedback from the market drives ideas and concepts for ongoing process improvement	Usability institutionalized

Because more companies intend to gain CMM certification (levels 3 and 4 are realistic), CMM model can be also used as interesting opportunity for introducing usability into organizations. Obtaining certain CMM level in software industry is often more appreciated than having ISO 9000 certificate in other branches. Usability engineering can be thus introduced to routine development process at smaller costs and less effort, if one of quality programs was accepted as a corporate goal.

Conclusions

In recent years we can observe reduced interest of software companies in certification of products, also because certification procedure has to be paid every time a new version of a product is released (once a year in many cases). For this reason much more cost-effective is the certification of manufacturing process, which can be used for a number of products, for several years.

This shift of interest from pure usability evaluation to more complex quality assurance services shows that usability consultants have to apply a broader view, including also other initiatives for software process and product improvements, taking place in software engineering community[2]. These initiatives for software quality improvement usually include some part on product usability, what gives a good chance to address usability issues in product concept, testing programs and other areas of the project.

[2] For instance: ESPI – European Software Process Improvement Foundation, INSPIRE – European Project "Initiative for Software Process Improvement", Bootstrap – European assessment approach, TickIT – ISO 9000-3 certification scheme, SPICE – ISO standard for Software Process Assessment

In the future isolated usability techniques and product-oriented evaluation methods will probably suffer from lack of interest, if they are not fully incorporated into the software development process. Project managers are interested in benefits from usability engineering, but the outcome must be realistic and usability must be seen as a tool for building competitive edge for the product – or for the whole company.

Supporting shift in usability consulting from product-oriented to process-oriented services can also help to overcome some common problems like:

- separating user interface design and product usability concept from programming technology,
- including usability as routine and typical part of testing programme,
- specifying the context and definition of usability at each stage of software product lifecycle, and people responsible for this part of quality,
- costs of building a system for systematic monitoring and evaluation of quality characteristics for software products; reusability of such system could help also in taking marketing decisions, benchmarking studies or customization of product for different groups of users.

References

Herbsleb J., Zubrow D., Goldenson D, Hayes W., Poulk M., 1997. Software Quality and Capability Maturity Model. In: Communications of the ACM, June 1997, Vol. 40, No. 6, pp.30-40.

Bevan. N, 1997. Planning and Inplementing User-Centered Design. NPL, Teddington.

IBM, 1992. Object-oriented user Interface Design: IBM Common User Access Guidelines. Que Corporation, Carmel, IN,.

ISO 9000-3, 1991. Guidelines for the Application of ISO 9001 to the Development, Supply and Maintenence of software. Intrenational Standard.

ISO 9241-11, 1995. Ergonomic Requirements for Office Work with Visual Display Terminals. Part 11, Guidance on specifying and measuring usability, Draft International Standard.

ISO DIS 13407, 1997. Human-Centered Design Processes for Interactive systems.

Microsoft Windows, 1993. Graphical User Inreface Design Gudelines. Microsoft Publishing, Redmont, VA.

Newman W.M., Lamming M.G., 1995. Interactive Systems Design. Addison Wesley, New York.

Oppermann R., Murchner B, Reiterer H., Koch M., 1992. Software-ergonomische Evaluation. Die Leitfaden EVADIS II. de Gruyer, Berlin.

Preece J., 1994. Human-Computer Interaction. Addison Wesley, New York.

Ravden J.S., Johnson G.I. 1989. Evaluating Usability of Human-Computer Interfaces. A Practical Method. Wiley, New York.

Reiterer H., 1993. The Development of Design Asid Tools for Human Factors Based User Interface Design. IEEE International Conference on Systems, Man and Cybernetics, 1993, pp. 361-366.

Sikorski M., 1996. Quality Models in Usability Evaluation of Business Management Software. In: Proceedings of 8th European Conference of Cognitive Ergonomics. Granada, Spain. 10-13 Sept., 1996. 121-124.

WORKSHOP PARTICIPANTS

Nigel BEVAN
National Physical Laboratory
Teddington, Middlesex, TW11 OLW, **UK**
e-mail: *Nigel.Bevan@npl.co.uk*
tel. +44 181 9436306 fax. +44 181 9436306

Edmund BUCHBINDER
Bűro fur Arbeits- und Organisationspsychologie
Gutzmannstrasse 38, 14165 Berlin, **Germany**
e-mail: *bao@bao.de*
tel. +49 30 84509303 fax. +49 30 84509304

Bożena CIERLIK
Human Factors Research Group, University College,
Donovan's Road, Cork, **Ireland**
e-mail: *bozena@ucc.ie*
tel. +353 21 902412 fax. +353 21 270439

Mili DOCAMPO RAMA
Institute for Perception Research (IPO)
Center for Reseach on User-System Interaction
P.O. Box 513, 5600 MB Eindhoven, **The Netherlands**
e-mail: *docampo@ipo.tue.nl*
tel. +31 40 2475239 fax. +31 40 2431930

Nigel CLARIDGE
Nomos Management AB
Svardvagen 3A, Box 119 , S-18212 Danderyd, **Sweden**
e-mail: *nigel.claridge@nomos.se*
tel. +46 1 7536220 fax. +46 1 7536793

Morten FJELD
Swiss Federal Institute of Technology,
Institute for Hygiene and Applied Physiology,
Clausiusstrasse 25, CH-8092 Zürich, **Switzerland**
e-mail: *fjeld@iha.bepr.ethz.ch*
tel. +41 1 6323983 fax: +41 1 6321173

Sissel GUTTORMSEN-SCHÄR
Swiss Federal Institute of Technology,
Institute for Hygiene and Applied Physiology,
Clausiusstrasse 25, CH-8092 Zürich, **Switzerland**
e-mail: *guttormsen@iha.bepr.ethz.ch*
tel. +41 1 6322531 fax: +41 1 6321173

Marc HASSENZAHL
Bűro fur Arbeits- und Organisationspsychologie
Gutzmannstrasse 38, 14165 Berlin, **Germany**
e-mail: *bao@bao.de*
tel. +49 30 84509303 fax. +49 30 84509304

Gerard HOLLEMANS
Institute for Perception Research (IPO)
Center for Reseach on User-System Interaction
P.O. Box 513, 5600 MB Eindhoven, **The Netherlands**
e-mail: *hollemans@ipo.tue.nl*
tel. +31 40 2475244 fax. +31 40 2431930

Jurek KIRAKOWSKI
Human Factors Research Group, University College,
Donovan's Road, Cork, **Ireland**
e-mail: *jzk@ucc.ie*
tel. +353 21 902636 fax. +353 21 270439

Marian KURAŚ
Cracow Academy of Economics
Dept. of Informatics
ul. Rakowicka 27 31-150 Cracow, **Poland**
e-mail: *eikuras@cyf-kr.edu.pl*
tel. +48 12 616 7235 fax: +48 (12) 421 0353

Elly LAMMERS
Vrije Universiteit, Computer Science Department
De Boelelaan 1081A 1081 HV Amsterdam, **The Netherlands**
e-mail: *elly@cs.vu.nl*
tel. +31 20 4447764 fax. +31 20 6441746

Martin MAGUIRE
HUSAT Research Institute (Loughborough University)
The Elms, Elms Grove, Leics, LE11 1RG, **UK**
e-mail: *m.c.maguire@lboro.ac.uk*
tel. +44 1509 611088 fax. +44 1509 2344651

Reinhard OPPERMANN
German National Research Centre for Information Technology GMD-FIT
D-53754 St. Augustin, **Germany**
e-mail: *oppermann@gmd.de*
tel. +49 2241 142703 fax. +49 2241 142065

Jochen PRŰMPER
Bűro fur Arbeits- und Organisationspsychologie
Gutzmannstrasse 38, 14165 Berlin, **Germany**
e-mail: *bao@bao.de*
tel. +49 30 84509303 fax. +49 30 84509304

Matthias RAUTERBERG
Swiss Federal Institute of Technology,
Institute for Hygiene and Applied Physiology,
Clausiusstrasse 25, CH-8092 Zürich, **Switzerland**
and
Institute for Perception Research (IPO)
Center for Reseach on User-System Interaction
Den Dolech 2, 5612 AZ Eindhoven, **The Netherlands**
e-mail: *rauterberg@ipo.tue.nl*
tel. +31 (0)40 24752 42   fax: +31 (0)40 24319 30

Marcin SIKORSKI
Technical University of Gdansk, Ergonomics Dept.
ul. Narutowicza 11/12, 80-952 Gdańsk, **Poland**
e-mail: *msik@zie.pg.gda.pl*
tel. + 48 58 3471812   fax. + 48 58 3471861

Stanislaw SZEJKO
Technical University of Gdansk, Dept. of Applied Informatics
ul. Narutowicza 11/12, 80-952 Gdańsk, **Poland**
e-mail: *stasz@pg.gda.pl*
tel. + 48 58 3471118   fax. + 48 58 3472727

Gerrit van der VEER
Vrije Universiteit, Computer Science Department
De Boelelaan 1081A 1081 HV Amsterdam, **The Netherlands**
e-mail: *gerrit@cs.vu.nl*
tel. +31 20 4447764   fax. +31 20 6441746

Paulus H. VOSSEN
Fraunhofer-Institut für Arbeitswirtschaft und Organisation IAO
Nobelstr. 12 D-70569 Stuttgart **Germany**
e-mail: *Paulus.Vossen@iao.fhg.de*
tel. +49 711 970 2315   fax. +49 711 970 2300