13

Domain specific evaluation during the design of human-computer interfaces

Magnus Lif and Bengt Sandblad
Uppsala University, Center for Human-Computer Studies,
Lägerhyddv. 18, S-752 37 Uppsala, Sweden.
tel. +46-18-183321 (sekr), fax. +46-18-187867,
magnus.lif@syscon.uu.se, bengt.sandblad@cmd.uu.se

Abstract

During our work within highly domain specific development projects we have noticed that a user centred approach in design does not necessarily lead to a user interface that is "optimal" for the skilled end-user. There is a need for a evaluation method that identifies both general and domain specific usability problems, identifies usability problems concerning both efficiency in daily use and ease of learning, can be included early in the user centred design process, is cost efficient and easy to use.

This paper presents an evaluation method that identifies general usability problems by using a variant of heuristic evaluation, and domain specific usability problems by evaluating the interface together with the end-users. The heuristics have been adjusted to fit evaluation of interfaces for skilled users with great domain specific knowledge. The method is to be included early in the user centred design process as a tool to guide the development of the interface towards a design that is efficient for the end-user in his/her specific domain.

Keywords

Domain specific, evaluation, skilled users, efficiency in daily use, heuristics, user centred

1. INTRODUCTION

There is an increasing demand for methods of including domain knowledge in the design of user interfaces. Neisser (1976) argues that instead of spending too much time modelling in laboratories to come up with good models of human-computer interaction it would be better to go out and study how people actually work in the real world. Cognitive ergonomic research claims that it is important not only to analys the user interface but also the domain beyond the visible interface (Fischer, 1993).

Norman and Draper (1986) suggest a user-centred approach to information system development in order to capture domain knowledge. In the past few years, this approach to design, in different shapes, has become quite common in system development projects. In user centred design there are usually two groups involved – developers and end-users. The developers are often software engineers with limited domain and usability knowledge, while the end-users are domain experts that often lack computer and usability knowledge.

A. Sutcliffe et al. (eds.), *Domain Knowledge for Interactive System Design*

The type of work we mainly study is administrative case handling work performed by **skilled professionals**, that only use and appreciate an information system as long as it efficiently supports the main purpose of the task, e.g. case handling. In some of the system development projects that we have been involved in, we have noticed that a user centred approach does not necessarily lead to a user interface that is "optimal" for the end-users. In these system development projects, the user interface was created during collaborative sessions by users and developers, during which the users expressed their requirements on the interface, in domain specific terms, and the developers interpreted these requirements into computer terms. The resulting interface was then realised as paper mock-ups and later in the development process implemented using prototyping tools. These prototypes were then tested by the users and new requirements on the interface were put forward. This iterative process was continued during the whole development process.

When developing a user interface with a user centred approach one would think that all domain specific usability problems would be avoided since the users continuously evaluate the prototypes. However, when developing systems for skilled end-users, this is not often the case. One reason for this is that it is difficult for the users to express their domain knowledge in a useful way (Whitefield, Wilson, and Dowell, 1991). When evaluating an information system it is important to capture the domain specific usability problems. The evaluation should focus on the given system and the intended goals of the system (Whitefield et al., 1991). Another drawback with these development projects is the lack of usability knowledge. Futhermore, there is a conflict between skilled and novice users. For skilled end-users, it is important that the computer support is efficient in daily use, while for novice users, it is more important that the computer support is easy to learn. There are several examples of guidelines for skilled users conflicting with guidelines for novice users. For a novice user, it is important to minimise the number of items of information on the screen (Nielsen, 1993). However, a skilled user requires and is able to scan a lot more information if it always has the same spatial location on the screen (Nygren, Allard, and Lind, 1996). Today's methods often focus on usability problems concerning ease of learning and not on efficiency in daily use. It is also important that the evaluation can be perfored early in the development process (Whitefield et al. 1991). It is then easier to guide the design towards a user interface that is efficient for the end-user.

There is a need for an evaluation method that fulfils the following criteria:

- Identification of both general and **domain specific** usability problems.
- Identification of usability problems concerning both **efficiency in daily use** and ease of learning.
- Capable of introduction early in an iterative design process.
- Cost efficient and easy to use.

2. EXISTING EVALUATION METHODS

Many methods for evaluating user interfaces have been published. They all have their advantages and disadvantages but it is not always obvious whether they are applicable in a specific system development project or not. In this paper we separate the evaluation methods into, usability testing methods, i.e. where users are involved, and usability inspection methods, i.e. where users are not involved.

A traditional method of user testing is **performance measurement** (Nielsen, 1993). The purpose of this is to measure whether a usability goal is reached or not. User performance is almost always measured by having a group of test users perform a pre-defined set of tasks while collecting data on errors and times. The tests are usually carried out in a laboratory. With such a test many usability problems will be found. One advantage of this test method is that the result is given in hard numbers which makes comparison of different design solutions easy. Unfortunately, in most system development projects there are not enough time, money or laboratory expertise to use this kind of method. Another problem with laboratory tests is that they are difficult to perform early in the design process since the tests demand a running prototype and require a reasonably complete database. It is also not always easy to measure more abstract goals using this method.

Questionnaires (Nielsen, 1993) are especially useful for issues concerning users' subjective satisfaction and possible anxieties. Questionnaires may be distributed to many users and is an inexpensive survey method. A drawback of this method is that end-users are not able to tell how they will interact with the system when they are skilled. Therefore it does not identify domain specific usability problems concerning efficiency in daily use.

Thinking aloud is another method that involves users (Nielsen, 1993). With this method the users verbalise their thoughts while using the system. Through this test, users let the usability expert understand how they view the computer system. This is an inexpensive test that identifies users' misconceptions of the system. Drawbacks with this method include that it is not very natural for users to think out loud and it is hard for expert users to verbalise their decision process. Expert users execute part of their work automatically (Schneider and Schiffrin, 1977; Shiffrin and Dumais, 1981). Therefore, it is difficult to capture domain specific usability problems concerning efficiency in daily use.

One method that includes users, developers and usability experts, and may be carried out early in the design process is **pluralistic walkthrough** (Nielsen and Mack, 1994). In this method, representatives from the three categories meet and discuss usability problems that are associated with the dialogue elements in different scenario steps. In pluralistic walkthrough the main focus is on how users react in different situations, e.g. a user may claim that, in a certain situation, he or she would "Hold down the shift key while pressing Enter". Pluralistic walkthrough is an effective method in evaluating the learnability of a user interface. It is however not as effective when evaluating an interfaces efficiency in daily use since the users are not able to predict how they will interact with the system when they are skilled.

There are also several different inspection methods available. One such method is **cognitive walkthrough** (Nielsen et al., 1994). With this method an evaluator examines each action in a solution path and tries to tell a credible story describing why the expected user would choose a certain action. The credible story is based on assumptions about the users background, knowledge and goals, and on understanding the problem solving process that enables a user to guess the correct action. The cognitive walkthrough is an inspection method that focuses on evaluating a design for ease of learning, particularly by exploration. Therefore, this method is also not applicable regarding inspection of interfaces for skilled users. Also, this method does not capture many domain specific usability problems, due to the evaluator's limited domain knowledge.

Another inspection method is **heuristic evaluation** (Nielsen et al., 1994). The evaluator uses sets of guidelines, i.e. heuristics, and compares those with the interface. The heuristics form a checklist that the evaluator uses during his work. It is easy to learn and inexpensive to use. With heuristic evaluation it is possible to identify many usability problems and it is possible to evaluate early on in the design phase. A drawback is that evaluators using this method seldom manage to identify domain specific usability problems due to lack of domain knowledge. Some kind of heuristic evaluation would however be useful when identifying

general usability problems in an interface. The heuristics that Nielsen suggests (Nielsen, 1993) work for a broad range of interfaces but have an emphasise on ease of learning. The heuristics are not "optimised" for identification of usability problems concerning efficiency in daily use.

In this paper we present an evaluation method that fulfils the previously mentioned demands by using a variant of heuristic evaluation to identify general usability problems and by evaluating the interface together with users to identify domain specific problems. This method has been developed and tested in different system development projects.

3. THE EVALUATION METHOD

Identification of general and domain specific usability problems.

Our field studies have shown that there are some usability problems that are possible to discover for a usability expert without specific domain knowledge, i.e. general usability problems. To discover domain specific usability problems the evaluator needs knowledge about the end-users work. This is often a problem as the evaluator seldom has enough domain knowledge. With this method it is possible to identify both general and domain specific usability problems using sets of heuristics. We have made a clear distinction between general and domain specific heuristics and the method consists of two different evaluation parts: One in which the evaluator inspect the interface with the purpose of finding general usability problems and one where the purpose is to identify domain specific usability problems. In the latter part of the evaluation, domain experts, i.e. end-users, are involved. One reason for this distinction is to make the method more cost efficient. Involving users during the evaluation is expensive why it is cheaper to identify the general and the domain specific usability problems separately.

Identification of usability problems concerning both efficiency in daily use and ease of learning.

The heuristics used in this method are based on standards (e.g., ISO, 1995) and controlled psychological experiments, (e.g., Nygren et al., 1996). Other bases for these heuristics are field studies of work activities (e.g., Nygren and Henriksson, 1992) and participation in development projects (e.g., Borälv, Göransson, Olsson, and Sandblad, 1994). Another important basis for the selection of relevant heuristics is our experience gained during the development of another method, the ADA method (Åborg, Sandblad, Strigård, and Nygren, 1994), for evaluation of usability problems in occupational health studies. In the ADA method the importance of different heuristics in this context has been evaluated. The heuristics of the method presented in this paper have been found useful in many projects that we have been involved in. During these projects the main focus has been to identify usability problems in interfaces for skilled users in administrative work. Therefore the heuristics are adjusted for that.

Capable of introduction early in an iterative design process.

Normally, development of computer support can be divided into three different processes: analysis, design and construction. The analysis process includes analysis of the organisation in the domain, the information that is used in different work tasks and analysis of the information utilisation (Gulliksen, Lind, Lif, and Sandblad, 1995). During the design process a prototype is developed, based on results from the analysis. This evaluation method is meant to be used as a tool to guide the development of the interface towards a design that is efficient

for the user. Since it is easier to make corrections if usability problems are discovered early in an iterative design process, this method should be applied from the beginning of the process. The result of the design process is an executable prototype. The iterative development process should continue during construction of the computer support.

Cost efficient and easy to use.
When using this method there is no need for expensive laboratory equipment. The main cost is the evaluators and the users work time, but since this method is incorporated in a user centred design process, where users are already involved, there are few additional costs. Furthermore, the users engagement is not needed during the entire evaluation.

The heuristics and the method may be taught to a usability expert during a half-day tutorial and the evaluation sessions are also short.

3.1 How the method is used

The analysis of the domain, the users, and the technical environment is the basis for the design of the first prototype (Lif, Gulliksen, Lind, and Sandblad, 1996). This prototype may be a paper mock-up or a more complete prototype. In a user centred design process the users continuously evaluate the prototype during the iterative process. The domain specific evaluation described in this paper is a complement to the users evaluation and may be used in parallel to it. See Figure 1.

After every evaluation, the prototype may be changed by the developers, according to the evaluation results, and then handed over to the users for testing. A new evaluation process may then start. This iterative process continues until the application is implemented and incorporated in the working environment.

The domain specific evaluation of a user interface is divided into four steps:

1. Identify general usability problems
2. Identify domain specific usability problems
3. Document the findings
4. Give feedback to the developers

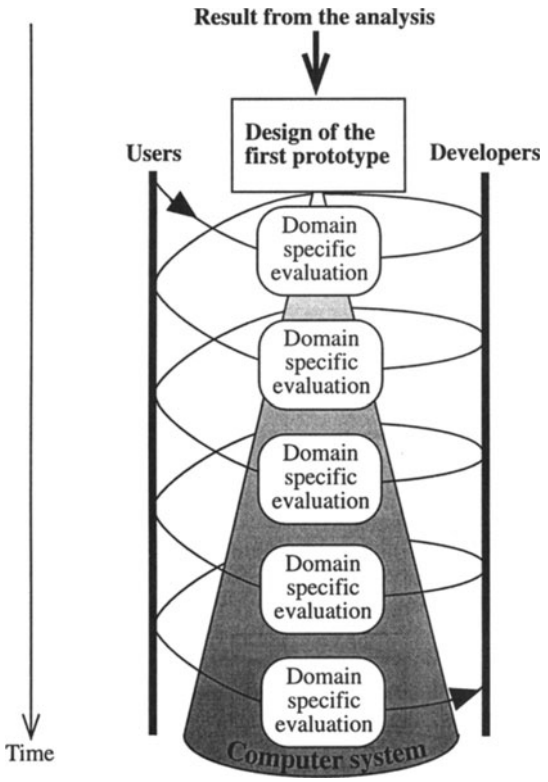

Figure 1 Domain specific evaluation in a user centred design process.

Who is the evaluator

Heuristic evaluation is more effective if carried out by a usability expert than by a non-expert (Desurvie, Kondziela, and Atwood, 1992). To be able to use this method the evaluator needs usability knowledge to understand the contents of the different heuristics and some domain knowledge. It is necessary that the evaluator is able to cooperate with both users and developers.

The steps involved in the method

Each domain specific evaluation is performed in four different steps.

Step 1. Identify general usability problems.

In this step the evaluator identifies general usability problems. This is done in a similar way to heuristic evaluation (Nielsen et al., 1994). The heuristics form a checklist that the evaluator uses during his work. During the inspection the evaluator comes up with an opinion about what is good and what is bad about the interface.

The heuristics that Nielsen proposes differ slightly from the ones used in this method. We have chosen to separate the general from the domain specific heuristics and this method is intended for evaluation with more emphasises on efficiency in daily use.

During our work on evaluation of user interfaces in different projects we have found the following general heuristics important and useful.

1. Obvious functionality and layout
A workspace consists of a collection of information entities, assembled to fit the needs of a specific work situation (Lif et al., 1996). When working with computer support it should be clear which the main entities are and how they are related to each other. A workspace should normally consist of a few main information entities, e.g. an area for documents, an area for input, an overview, a row with icons and a menu. These entities should be distinct and have permanent locations on the screen.

2. Orientation and navigation
It is important that the user is well orientated in the system i.e. knows where he/she is in the system, how he/she got there and how to precede. This may be done by showing an overview at the same time as showing details or by presenting the way a user has "walked" through the system (Henderson and Card, 1987). It must also be clear if there is information that is not visible and how to make this information visible.

3. Give data a specific location
A skilled user is able to quickly scan a lot of frequently used information, if it always has the same spatial location on the screen (Nygren et al., 1996).

4. Consistency
The system should be consistent in functionality and layout (Foley, 1990). This will encourage exploratory learning and is also necessary for efficiency in daily use.

5. Minimise the users system related input
Moving around in the system takes a considerable amount of time. Avoid this by using broad instead of deep menus and minimise the number of steps in a dialogue. Input with a mouse is normally slower than input from a keyboard. Therefore it is important to use shortcuts and default options (Nielsen, 1993). Use codes to show if it is possible to chose a certain alternative or not.

6. Give distinct feedback
Give distinct feedback from the system in reaction to the user's input within reasonable time (ISO, 1995). Give the user information if the waiting time is longer.

7. Allow and prevent errors
Allow the user to make mistakes (ISO, 1995). Support "undo" and "redo" and give constructive error messages. Minimise probability of occurrence of errors.

8. On-line help
Give on-line help on the user's initiative (ISO, 1995). This help should be easy to access and understand. If no other solution is possible, give help at the initiative of the system.

9. Controllability
The user must be able to control the system and to carry out operations in different orders (ISO, 1995).

10. Readability
Use fonts, sizes of characters, colours, shapes and forms in a way that makes the information readable on the screen.

11. Discrete layout
Do not overdo it. Use colours, fonts and decorations with care. Do not use too many colours, fonts, frames, boxes and lines in a user interface. This may interfere with the coding of information in the interface. Use discrete colours for unimportant information such as the background.

This list of heuristics serves as a checklist for the evaluator. This helps the evaluator to control that all heuristics have been considered during the evaluation. It is however not necessary for the evaluator to follow the checklist in a specific order during the evaluation, but it is important that all aspects are documented (see step 3).

During a typical inspection the evaluator will study the prototype carefully and try to identify the advantages and disadvantages of the different design solutions. It is important to continuously write down both the opinion and, if it is not obvious, reason for the opinion. The time this takes differs, depending on the complexity of the prototype and the evaluators experience, but it should normally not take more than two hours.

Step 2: Identify domain specific usability problems
This step is a bit more complicated since it is necessary to involve users. During our work we have found that one efficient way to evaluate the interface is with a group of users. The evaluator leads a discussion, based on the prototype, displayed either on paper mock-ups or on a screen. The evaluator should, together with the users, identify the different work tasks and find out if all information needed is visible while performing each task, which information is needed often or seldom, how the decisions are made etc. The checklist below is used to guide the discussion.

This step demands a lot from the evaluator. The person leading the discussion has to be able to understand the users terminology, and keep the discussion on track without inhibiting the free flow of ideas and comments. He must also be observant so that all members in the group have the opportunity to express their opinion. Another problem is that it may be difficult for the users to understand how the system will function in their normal work environment. Therefore the discussion need to be concentrated around the work tasks and the domain specific problems and not on more general problems where human factor knowledge is needed.

Domain specific heuristics needed to be considered:

1. Work related layout and functionality
The workspace i.e. the screen layout should fit the needs of a specific work situation. The domain specific elements must be relevant for the work and the information should be logically grouped (ISO, 1995). Use metaphors that are familiar to the user. The metaphors may be found by studying the end-users current work (Nielsen, 1993).

2. Simultaneous presentation of information
The user should not have to remember information from one part of the dialogue to another. If possible, show all information that is needed in a work task on the screen simultaneously (Lind, 1991). The information that is used most often should be visible at all times.

3. Emphasise important information
Use fonts, sizes of characters, colours, shapes and forms to emphasise important information (Nygren, Johnson, and Lind, 1992). Do not emphasise unimportant information. For a skilled user, data is more important than labels etc.

4. Use visual cues for task related status
Show task related status with highlights, character styles, colour codes and by using dedicated positions for application data (Nygren et al., 1996). This will decrease the amount of time a skilled user needs to find information.

5. Use shortcuts and default values
Use shortcuts and default values to minimise the user's input (Nielsen, 1993). Sequences of inputs should be possible to perform in a logical order.

6. Speak the users language
The system should speak the users language with words, phrases, symbols, icons etc. (Nielsen, 1993). Avoid system-oriented terms.

The results from this discussion is documented in the checklist (see step 3). The evaluator should also take notes on problems and questions that occur during the session that are not included in the checklist.

Step 3. Document the findings
The results from the two previous steps are documented according to the structure of a checklist from which it is clear which heuristics are inapplicable, applicable and met, or applicable and not met (ISO, 1995). It is also important to describe the reasons for opinions if they are not obvious.

In the example below we will look at the documentation of one domain specific heuristic in the checklist, no 2. Simultaneous presentation of information. In this documentation we use **?** for non applicable heuristic, **+** for applicable and met and **-** for applicable and not met. The example is from a system development project at the Swedish National Tax Board. See Figure 2.

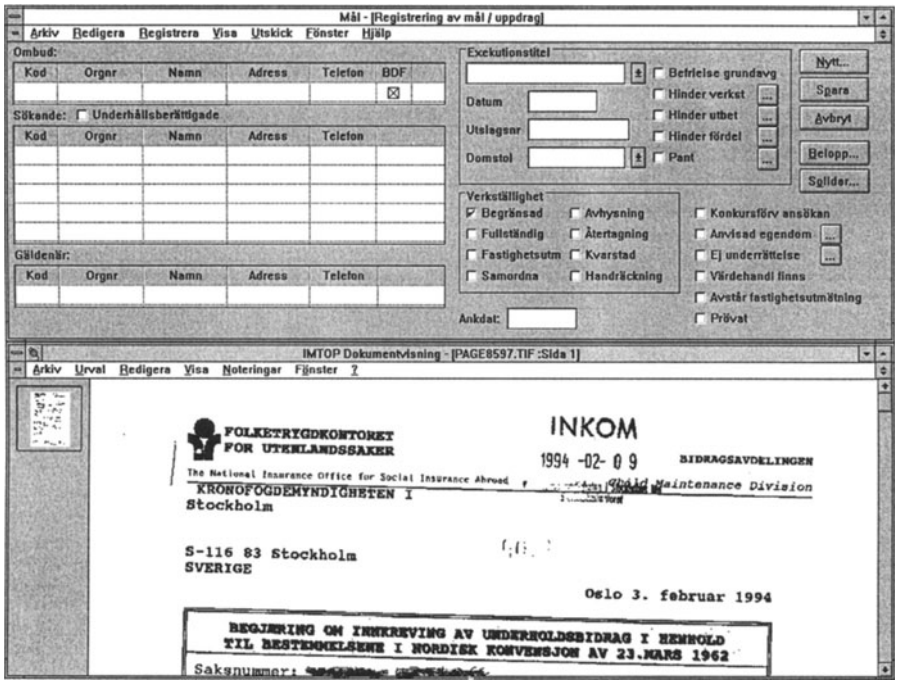

Figure 2 Screen layout where the task is to register an application.

2. Simultaneous presentation of information
+ In the work task, all input fields on the screen need to be filled in. No unnecessary fields are visible.
- The input field for entering the "Adress" is too small.
- Several input fields needed to be filled in is not visible, e.g. "Belopp..." and "Solidar..."
- Only part of the scanned document is visible. Scrolling is needed which is time consuming.
? It is not possible to say if the user needs to see different information in the document, simultaneously.

Documentation is done in a similar way for all heuristics.

Step 4. Give feedback to the developers

In this step the evaluator discusses the results of the evaluation with the developers and the users. To make this discussion efficient, all participants should have studied the documentation in advance.

During this step it is advantageous if the participants first concentrate on the major usability problems that have been identified during the evaluation and then continue with the minor problems. This ordering of problems is prepared by the evaluator that will lead the discussion. It is important that all three groups participate, since they all have their own knowledge to contribute. The users are experts on the domain, the developers know the technical limitations and if a design solution is possible or not, and the evaluator knows if a solution may lead to usability problems. Since design of interfaces always involves a lot of compromising, it is important that all participants are flexible during this discussion. It is then

possible for the whole group to find new and better design solutions. Support for this work may be found in e.g. the Design Rationale approach, where the main focus is on the documentation of the reasons behind design decisions (MacLean, Bellotti, and Young, 1990; McKerlie and MacLean, 1993)

When the last step is finished it is time for the developers to create a new prototype. The new prototype may then be evaluated in the same way. By continuing this iterative process the user interface will be developed.

4. DISCUSSION

This method has been used and evaluated during several highly domain specific projects concerning system development for administrative case handling work. With this method, we have discovered many, both general and domain specific usability problems early in the user centred design process. Thereby we have been able to improve interfaces at an early stage of development. Since the heuristics are adjusted to identify usability problems concerning efficiency in daily use, they have shown to be valuable when evaluating interfaces for skilled users.

In our work we have involved users that have already participated during the development process. This has the advantage that they know how the system is planned to work. The disadvantage is that they often are very fond of their own design solutions and not willing to make changes. We have also noted that evaluators having experience with this method are able to identify more important usability problems than evaluators without experience.

We are currently planning a more strict evaluation of this method and we will continue to develop the heuristics during our future involvement in system development projects.

This evaluation method is meant to be used **during** the design process. When evaluating a computer support that has been incorporated in a work environment for a longer period other methods, e.g. the ADA-method (Åborg et al., 1994), are more appropriate.

5. REFERENCES

Borälv, E., Göransson, B., Olsson, E., and Sandblad, B. (1994) Usability and Efficiency. The HELIOS Approach to Development of User Interfaces. In The HELIOS Software Engineering Environment (ed. U. Engelmann, F. C. Jean, and P. Degoulet), *Supplement to Computer Methods and Programs in Biomedicine*, **45**, 47-64.

Desurvie, H., Kondziela, J. and Atwood, M. (1992) What is gained and lost when using evaluation methods other then empirical testing. In *People and computers VII* (ed. Monk, Diaper and Harrison), Proceedings of the BCSHCI'92, Cambridge University Press, London.

Fischer, G. (1993) Beyond Human-Computer Interaction: Designing Useful and Usable Computational Environments. In *People and Computers VIII* (ed. J. Alty, D. Diaper and S. Guest), Proceedings of the BCSHCI'93 conference, Cambridge University Press, Cambridge, U.K.

Foley, J.D. (1990) Dialogue Design. In *Computer Graphics: Principles and Practise* (ed. J.D. Foley, A. Van Dam, S.K. Feiner and J.F. Hughes), Reading, Addison-Wesley, 391-431.

Gulliksen, J., Lind, M., Lif, M. and Sandblad, B. (1995) Efficient Development of Organisations and Information Technology - A Design Approach. In *Symbiosis of Human and Artifact.* (ed. Y. Anzai and K. Ogawa) Proceedings of the 6th International Conference on Human-Computer Interaction, HCI International'95, Pacificio Yokohama, Yokohama, Japan 9 - 14 July 1995, 951-6.

Hendersson, A., & Card, S. K. (1987) The Use of Multiple Virtual Workspaces to Reduce Space Contention in a Window-Based Graphical User Interface. *ACM Transactions on Graphics*, **5 (3)**, 211-43.

International Organisation for Standardization (1995) *ISO/DIS 9241-10 (Draft).* Ergonomic Requirements for Office Work with Visual Display Terminals (VDTs) - Part 10: Dialogue Principles.

International Organisation for Standardization (1995) *ISO/DIS 9241-11 (Draft).* Ergonomic Requirements for Office Work with Visual Display Terminals (VDTs) - Part 11: Guidance on Usability.

International Organisation for Standardization (1995) *ISO/DIS 9241-13 (Draft).* Ergonomic Requirements for Office Work with Visual Display Terminals (VDTs) - Part 13: User Guidance.

Lif, M., Gulliksen, J., Lind, M., and Sandblad, B. (1996), A Structural Approach to Prototype Design in Iterative System Development, *In preparation.*

Lind, M. (1991) Effects of Sequential and Simultaneous Presentations of Information. *CMD Report 19/91,* Center for Human-Computer Studies, Uppsala University, Sweden.

MacLean, A., Bellotti, V., and Young, R. (1990) What Rationale is there in Design? *Proceedings of the INTERACT '90 Conference on Human-Computer Interaction,* 207-12

McKerlie, D. & MacLean, A. (1993) QOC in Action: Using Design Rationale to Support Design. *INTERCHI '93 video program,* ACM: Amsterdam.

Nielsen, J. (1993) *Usability Engineering.* Academic Press, Inc. San Diego

Nielsen, J., Mack, R. L. (1994) *Usability Inspection Methods.* John Wiley & Sons, Inc

Neisser, U. (1976) *Cognition and Reality.* San Francisco: W H Freeman

Norman, D. A, Draper, S. (Eds.) (1986) *User Centred System Design.* Lawrence Erlbaum Associates, Inc., Hillsdale: New Jersey.

Nygren, E., and Henriksson, P. (1992) Reading the Medical Record I. Analysis of Physicians Ways of Reading the Medical Record., *Computer Methods and Programs in Biomedicine.*, **39**, 1-12.

Nygren, E., Allard, A., and Lind, M. (1996) *Skilled Users Interpretation of Visual Displays.* Submitted for Publication.

Nygren, E., Johnson, M., Lind, M. and Sandblad, B. (1992) The Art of the Obvious. *Proceedings of CHI '92,* Monterey: California, May 1992.

Schneider, W. and Schiffrin, R. M. (1977) Controlled and automatic human information processing: I., In *Psych. Rev.,* **84**, 1-66.

Shiffrin, R. M. and Dumais, S. T. (1981) The Development of Automatism. In *Cognitive Skills and their Acquisition.* (ed. J. R. Anderson), Hillsdale, NJ, Erlbaum.

Whitefield, A., Wilson, F., and Dowell, J. (1991) A framework for human factors evaluation. In Behaviour & information technology, **10 (1)**, 65-79

Åborg, K., Sandblad, B., Strigård, W., and Nygren, E. (1994) A practical method for evaluation of human computer interfaces. In *Book of short papers. 4th international scientific conference "Work with display units".* 2-5 October 1994, Milano.

6. BIOGRAPHY

Magnus Lif is a PhD student in Systems Analysis at the University of Uppsala. In 1994 he received a MSc in Engineering Physics. His present location is, the Center for Human-Computer Studies (CMD) at Uppsala University. His main research interests are HCI Design and Human Factors and he is involved in several applied research projects with the Swedish National Tax Board.

Bengt Sandblad received his Tech. Dr. in Systems Analysis from the University of Uppsala 1977. He has earlier been Associate professor and research leader at Uppsala University Data Center and presently at CMD, Uppsala University Center for Human-Computer Studies. He is a supervisor for HCI graduate students and lecturer in different undergraduate programs. Sandblad has been the chairman of SORA, the Swedish Operations Research Association and a board member of STIMDI, the Swedish Interdisciplinary Association for Human Computer Interaction. He has published a large number of papers in the fields of Systems Analysis, Computer Simulation and Human Computer Interaction.