TRITA-NA-D0006 • CID-72, KTH, Stockholm, Sweden 2000

How to make User Centred Design Usable

Jan Gulliksen, Ann Lantz and Inger Boivie

KUNGL TEKNISKA HÖGSKOLAN
Royal Institute of Technology
Numerical Analysis and Computing Science

TRITA-NA-D0006 • CID-72, KTH, Stockholm, Sweden 2000

How to make User Centred Design Usable

Jan Gulliksen, Ann Lantz and Inger Boivie

CID
Centre for
User Oriented IT Design

Jan Gulliksen, Ann Lantz and Inger Boivie

How to make User Centred Design Usable

Report number: TRITA-NA-D0006, CID-72

ISSN number: ISSN 1403-073X

Publication date: January 2000

E-mail of authors: Jan.Gulliksen@hci.uu.se, alz@nada.kth.se, inger.boivie@tietoenator.com

URL of authors: http://www.nada.kth.se/cid/

Reports can be ordered from:

CID, Centre for User Oriented IT Design
Nada, Dept. Computing Science
KTH, Royal Institute of Technology
S-100 44 Stockhom, Sweden
telephone: + 46 8 790 91 00
fax: + 46 8 790 90 99
e-mail: cid@nada.kth.se
URL: http://www.nada.kth.se/cid/

Contents

Preface

This CID report is a summary of the workshop “How to make user-centred design usable” held in Edinburgh, Scotland, on August 30-31, 1999. Attached are also the position papers submitted to the workshop.

After our previous workshop on ”User-centred Design in Practice – Problems and Possibilities”, arranged in conjunction with the Participatory Design Conference (PDC ’98) in Seattle, November 1998, we felt a bit frustrated that the main result of the workshop was a list of different problems and appalling examples from real life projects, for others to laugh at or to be disheartened by. We felt that there was a need for a more focused workshop that specifically addressed the problems identified in the first workshop. This report was produced to describe some of the workshop discussions and to give you the opportunity to read the papers submitted to the workshop. The overall result of this second workshop on user-centred design is the recognition of the importance of keeping up the discussion on how to facilitate user-centred design in practice in the future.

For more information see the web site for the INTERACT’99 workshop http://www.nada.kth.se/cid/interact99/workshop/index.html

The summary report has been submitted for publication in the SIGCHI Bulletin. All position papers have been reproduced with permission from the authors.

A short workshop summary was published in Stephen Brewster, Alison Cawsey and Gilbert Cockton (Eds.) Human-Computer Interaction INTERACT ’99 (Volume II), British Computer Society and IFIP.

Jan Gulliksen, Ann Lantz, Inger Boivie

Making User Centred Design Usable

A summary of the 1999 INTERACT workshop

Introduction

Few people would object to the necessity of involving users in the systems development process. In reality, however, user participation is often concentrated to the early phases of modelling and analysis, and usability activities are scheduled late in the project. The actual user interface design is often done with a minimum amount of communication with the users, in as little time as possible. Not because systems developers in general do not care for the users and usability, but because time schedules often are tight, communication is difficult and time consuming and usability aspects are simply not a priority.

How do we make user-centred design (UCD) support the user interface designers in their work and help them meet their deadline? To discuss these issues, the authors organised a two-day workshop at the INTERACT ’99 conference in Edinburgh, Scotland, August 30-31, 1999.

Eleven position papers were accepted and the workshop gathered fourteen participants from four countries. Below is a summary of the position papers.

The position papers were presented at the start of the workshop, and the remaining time was spent discussing matters as described below.

Previous workshop

The first workshop on “User-centred design in practice – Problems and possibilities” was held in conjunction with the 1998 Participatory Design Conference (PDC ’98) in Seattle, November 1998. The summary of this workshop was published in SIGCHI Bulletin [1] together with all the submitted contributions. This workshop illuminated the difficulties in adopting a fully user-centred design approach in practice and the need to spend more efforts on making the UCD process work better. This workshop specifically addressed:

- when and how to involve the user in a design and development process
- practical experiences of prototyping and video recording in the analysis, design and evaluation processes
- organisational obstacles to user-centred design
- the role of the UCD facilitator in the development process
- communication problems that occur when people with varied skills and expertise communicate with one another

Based on this workshop we felt the need to arrange a follow up workshop addressing further the need to make the user-centred design process more usable for its users.

Position papers

The position papers are summarised briefly below. The full versions of the papers are available on the workshop website:
http://www.nada.kth.se/cid/interact99/workshop/index.html

- *Easy-to-learn methods versus continuous learning in User Centred Design* – Bagger & Buur
  This paper discusses different teaching approaches in UCD – i.e. rigid methods versus continuous learning. The authors argue the advantages of modifying the methods in the teaching situation, so that they are focused on the concrete task at hand, over the teaching of the methods as such.

- *Intertwining Themes - Teaching Students How to Design New Technology for Actual Use and Bringing Design Issues to the Use Situation* – Eriksén
  The author brings up two questions in regards to UCD – ”How can UCD be taught?” and ”How can we bring UCD to the user, in the live use situation?” The author argues that in teaching UCD, you should focus on IT *in use.* The author also emphasises the impact of good IT management on usability.

- *Organisation and Communication for Usable UCD* – Gulliksen, Lantz & Boivie
  We discuss organisational aspects and communication aspects in a UCD process. How do we create a culture within an organisation that promotes the UCD approach? How do we communicate results from, e.g., HCI activities to the systems developers?

- *Delivering user-centred design tools to project teams with no usability specialists* – Harker
  This paper relates 15 years of experience of developing and teaching UCD tools intended for non-usability experts. The author points out the importance of simplicity in the tools, continuous training and adaptation of the tools to the situation at hand.

- *Making UCD possible as part of a structured design method* – Borup Harning
  The author describes a structured design process, where different types of requirements are mapped on user interface designs. The method uses different models of the design in different phases of the process, to visualise the design to the users.

- *Facilitating Communication during User-Centred Design* – Masoodian
  Communication breakdowns and the importance of facilitating effective communication in UCD projects are addressed in this paper. Videos are one way of improving communication between different members of a project.

- *A Case Study of User-Led Systems Design and Development in Healthcare Informatics* – Procter, Hartswood

This paper describes a user-led development project in healthcare informatics. The issues addressed were, among others, the form, content and timings of the contributions that users are able to make to design and development and the barriers to their effective involvement.

- *Creating a User-focused culture in an Internet company* - Rajkumar (not present at the workshop)
  Difficulties that are particular to a web development project are described in this paper. They are for instance, tight delivery deadlines, the success of sites despite poor usability and the wide range of users of Internet sites.

- *Using UCD for the Web - a case story* – Sørensen
  The author argues the importance of using boundary objects, for instance prototypes, for creating a common knowledge base in a project. Prototypes must, however, be ”packed” to suit the context of the different groups in the project.

- *On discerning users* – Thimbleby
  This paper makes three main points: a) problems with usability are cultural – usability is not glamorous for the manufacturers and thus ignored by them b) design tools that allow widespread collaboration should be available on the Internet and c) users understanding what usability is about is essential to successful user-centred design.

- *Making user-centred design usable* – Whitehand & Claridge
  A description of the successful use of scenarios and parallel design activities in a design process is provided in this paper.

Crucial Aspects

The title of the workshop was “Making User-Centred Design Usable”- but usable for whom? In the call for papers, we focused on the needs of the systems developers as being the primary ”users” of the user-centred design process. The workshop participants did, however, not share this standpoint. It was argued that no particular group can be singled out as being more important than the others are, the process must be usable for everyone involved. But, what is usable for one group may not be usable for others. Systems developers have other needs and requirements than users.

The participants were asked to name one aspect each, the one single aspect that they considered the most critical for the usability of the UCD process. The resulting list could be grouped into the below categories.

Communication

- Meet the users
- Learn more about human communication in daily life
- Support construction of shared understanding

Representations

- Understandable design representations - representations that make it possible for UCD people to communicate design with users and developers
- Equivalent design representations, i.e. different representations that convey the same information about an object but in different forms and terms

Process

- Good qualified, experienced usability experts
- Cultivate IT in use - see IT management as UCD. Start assessing the cost of not using IT the way it was supposed to be used.
- Ask the systems developers what they need
- Each organisation should specify its own UCD process - there are no silver bullets.

Attitudes

- Convey attitudes about UCD, not just UCD methods and tools
- Create a demand for usability guarantees
- Start with the management level - prove the business benefits
- The community of buyers, users and developers should acknowledge how much it costs to develop IT

The categories are not independent - making UCD more usable is, for instance, a matter of changing the process, but that requires a change in attitudes. Some of these items were further elaborated in group-discussions.

Communication

Communication is essential to the success of a systems development process, and a UCD process in particular. With the UCD approach, communication tends to become more complex. People of varying backgrounds and skills have to work together, each contributing his/her expertise in domains fairly unknown to the other participants in the process. One major aspect of the UCD process is therefore supporting communication between the different participants in the project. Who should talk to whom and where should the participants of the project meet? Should the systems developers/designers and the users communicate directly or should there be a UCD facilitator acting as a ”go-between”.

What are the responsibilities and tasks of a UCD facilitator? S/he needs to know about business targets and organisational goals. S/he also needs to know how to talk to the users and how to communicate results from the field to the developers. In addition to that, the UCD facilitator must be able to communicate technical limitations and other types of restraints to the users. In short, the UCD facilitator must understand and communicate the context of use as well as the context of development. In our discussions we agreed that

- the UCD facilitator must get a better understanding of how designers/developers work
- the designers/developers must meet the users.

Meeting, supports the construction of a shared understanding of, for instance, the context of use and the current status of technology. It might also lead to a shared understanding of different concepts contributed by the different groups of people involved.

Prototypes are a means for facilitating the construction of a shared understanding, as are other types of representations, for instance use scenarios, sketches, user data models and logical windows. The primary purpose of a prototype is to enable communication, in that it provides concrete features to discuss, rather than providing the one and only answer to the problem.

In his position paper, Masoodian argues that videos can be used to improve the effectiveness of group communication during a UCD process. In contexts outside organisational settings, the effectiveness of tools such as open-ended interviews and questionnaires is reduced. Instead, videos can be used. In this particular project, a video illustrating a use scenario was shown to potential users in a workshop. Their response was positive – they claimed that seeing the video helped them visualise the possibilities offered by the application. It also assisted the users in contributing new ideas to the design team.

We need tools or methods to help people see things in the same light. We also need improved theories about how people communicate and we need tools that facilitate communication on the field among different groups of people. Thus, we must work on the field and create examples that make communication more visible and more concrete.

Representations

Representations are context-dependent links between groups, abstractions with multiple faces. They can be used as platforms for creating a shared understanding of, for instance, a design solution or user requirements. Representations used in systems development must bridge the gap between the users and the developers/designers, as well as the gap between the HCI people or UCD facilitator and the developers/designers.

Representations of tasks and functionality are particularly important. Users think in terms of the tasks they perform, whereas systems developers need to think in terms of components or functions in the system.

People tend to group their tasks and create relationships between them. The relationship between user tasks can ultimately be broken down to the relationships between the functions in the systems. Thus we have a ”two-faced” representation of the tasks/functions in the system - one of which describes the tasks and the other face describes the functions. Sets of scenarios help the developers envision the system - all the tasks that are connected to a scenario can be listed.

Even if users think in terms of tasks, when performing the tasks they see the functionality provided by the system, so they shift between the different representations and the functionality approach can be shared with the systems developers.

Task modelling is, however, rather difficult, and not always the best approach. Tasks may be difficult to identify in, for instance, contexts that rely on information types rather than specific tasks. The same piece of information can be used in several different tasks. The system should basically provide the information and facilitate storing and retrieval of it.

Representations should allow its users to focus on specific tasks without losing touch with the context. Albeit being an old and well-used metaphor, the window is very useful in this respect. It is like a magnifying glass that you can move over a space to highlight particular items and features. Windows are useful for identifying what functions are required in a system. Let us assume we have the user tasks that we want to support in the future system. We can use these tasks to create windows, with which we can identify what functions are required for the particular task. These windows can then be evaluated together with users.

In his paper, Borup Harning describes a method for a structured design process, based on different representations of tasks and the information model. The representations evolve and change as the process moves along from the conceptual design phase to the physical design phase. In his model, Borup Harning uses forms to describe a user data model, containing the concepts and the pertaining information. The next step is to create logical or virtual windows and high level ”buttons”. In the last two phases of the design process, these representations are turned into the final user interface design, combining logical windows to dialogues, etc. The forms and the logical windows, and eventually the dialogues can be used to evaluate the design together with users.

Process

There is no silver bullet - no single process that will work in all situations. Or is there? Is the international standard on Human centred design process for interactive systems – ISO 13407 [2] the answer to life, to universe and everything? Or should we turn the design process upside down? Start the process with the design - to support the analysis of the problem at hand. Or do we simply need more and better formal methods?

Each software development organisation needs to design its own UCD process. It is only when faced with a particular problem that one can discuss and understand its solution fully. There are, however, different ways of tailoring the process.

- Use a standard process and adapt it to the context. When finished, reflect over what has been done, and express the real process in terms of the standard process.
- Pick and choose from a smorgasbord of tools, methods and techniques. Adapt the components to the context. Customise your process at every occasion. The question then is who should make up the ”prescriptions”, should the HCI community do that or should it be left to the individual organisation. How far should we go in describing what to choose from the smorgasbord and when to use each component?

Frameworks, such as ISO 13407, provide support but do not specify what to do and when. ISO 13407, moreover, stops with delivery; it does not cover the whole life cycle of a product or a system.

How do we take usability beyond the date of delivery? How do we cultivate IT in use? In some organisations, the systems developers spend a period of time working at the helpdesk. Another way is to involve the helpdesk people in the further development of a system or a product. Since the primary function of helpdesk is to support the users, helpdesk people generally get an accurate picture of the needs of the users. Another organisation had the systems developers check all the error reports. A majority could be traced back to requirements and usability problems. As a result, the requirement analysis phase was quite extended, whereas the development phase was significantly shorter than in other projects.

Whatever process or methods we use, a major problem is that of changing requirements. Partly due to the inability of those involved to identify and understand all the requirements in the early phases, partly because requirements do actually change in interaction with the evolving system or product. Therefore software must be allowed to evolve over time. In addition to that, development must be iterative, perhaps also incremental to reduce the risks involved.

Some participants felt that usability requirements are crucial to the usability of the resulting product or system. Such requirements must be specified in the early phases of the process, along with other types of requirements. But, in order for the systems developers to appreciate the importance of such requirements, special measures may be needed. One example was using a bonus system as an incentive for the systems developers to meet the requirements.

How do we make this complex process usable to the parties involved? As mentioned earlier, the needs and requirements differ between the groups. We need to address management as well as the systems developers when they need help with design decisions. There are different strategies, but some basic principles are:

- Management must be committed to UCD and usability. UCD specialists should be assigned to management position
- Managers must take the responsibility for the design solutions, good ones as well as bad ones
- Everyone involved must understand the full contents and impact of the delivery – not just the systems developers and a select group of users

Attitudes

In order to improve the usability in products, systems and gadgets, usability and UCD must be promoted within software development organisations. But no less important are the attitudes toward usability in the user communities.

”Unusability is discrimination”

H. Timbleby argues in his paper that unusability is a cultural problem. Usability is not glamorous for manufacturers and users do not or are not able to make choices informed by usability. Usability problems are blamed on the users - the users become ”fashion victims” forever buying into new, and ”better”, systems or upgrades in the futile hope of solving their problems with using their current system. The manufacturers, on the other hand, are forever improving performance and adding features. In this way they can put off the point in time where they have to start improving the usability of the existing features. The user community gets entangled in feature stepping – i.e. each individual user wants the best and upgrades his/her software to get all the latest features, thereby forcing other users to upgrade to yet another version with even more features. Manufacturers are well aware of and encourage this ”upgrade habit”.

Another problem is that usability is not always a selling point in consumer goods. The decisions a consumer makes in the buying situation are not governed by aspects such as usability and task-fitness, but by simple factors like brand name, colour and the number of buttons. So the manufacturers can simply ignore usability with impunity.

It is time to stop blaming the users for usability problems. Usability should be a legal requirement - if the manual does not describe the use of the system correctly the consumers should get their money back. H. Thimbleby suggests a Usability Response Team that would evaluate usability problems and put pressure on manufacturers to produce usable products.

What attitudes promote a usability focus and the UCD approach in systems development?

As mentioned earlier, it is essential that users and systems developers do get together. The systems developers or the UCD facilitator must get out to the people that actually work with the system or product, the real users. Never rely on management to describe the work practice correctly. The results of a UCD process are very much affected by what users you involve in the process. Apple, for instance, asked about feedback from those that bought the iMac, but not from people that did *not* buy it.

On the other hand, when the systems developers and the users do meet to evaluate design solutions, there is the problem of the "My baby syndrome". Negative criticism provided by the user is taken as a personal affront by the systems developer. It may be easier if the process were more ”mechanical” and the evaluations were a natural and less dramatic part of it. It is also important to make the iteration cycles short. The "My baby syndrome" gets worse with the amount of time and effort spent on producing a design solution.

Dissemination

Teaching user-centred design to students is one way of increasing the knowledge in the area but a slow one. It is impossible to teach everybody everything - nobody can do everything! At the workshop, the discussion was limited to the training of practitioners – i.e. how do we teach user centered design to those who are directly involved in the process, in real projects.

The training must focus on explaining essential aspects, helping people understand the basic principles and how to perform activities in practice. The objective of the training is to change peoples' attitudes about how they do things, help them think in new ways about what they do and how.

The training process is not independent of management. It is also strongly related to the organisation structure, as well as to expectations. Practitioners need, above all, a framework or a structure and principles – but they must also be made aware of the fact that these exist. Understanding of any kind involves recognising what opportunities are available and identifying and conceptualising things already known but yet not made aware.

”Ownership” is essential in a learning process. Having the participants work with problems from their own projects provides practical experience, creates awareness and facilitates the reflection of what has been done. When successfully employed, such an approach may result in the developers taking that extra responsibility for the user centered design activities in their project.

Conclusions and discussion

The purpose of the workshop was to discuss how the UCD process as such can be made usable to its “users”. How does, for instance, the UCD process best support the systems developers in their attempts to reconcile changing user requirements and designing the best possible user interface with tight deadlines? And how do we best support the users in their efforts to identify and express their own needs?

Our discussions resulted in, among other things, the below conclusions:

- Usability must be made an important issue at the top of an organisation – management must be committed to usability.

- Organisations must design their own UCD processes.

- Everyone involved needs to understand the full scope of the project and the problems that are inherent in the process – the UCD facilitator needs to understand the designers/developers and their problems, the designers/developers must understand the work situation and the problems of the users, the users must have a fair grasp of the technical and economical constraints of the project.

- Systems developers and designers should meet the users, without ”go-betweens” – *everyone* involved in the project should see the system or product in real use.

- Effective representations are essential for the success – representations facilitate communication and the building of a shared understanding. Good representations need to have multiple ”faces” and to be ”packed” to suit the different needs of the different groups involved.

The starting point of the workshop discussions was the list of crucial aspects falling into the categories Communication, Representations, Process and Attitudes as described above. These categories are, however, inter-related to one another where communication is an overall theme. Communication skills are critical for the success of the project. Good design practice may simply be about the ability to listen. The main purpose of the process is to support successful communication. Likewise is the main purpose of using representations and of promoting appropriate attitudes to facilitate communication.

The process provides a structure for who should talk to whom, about what and when. With representations we can communicate the results of such activities to the people within the project as well as to people outside the project. Attitudes are essential for the success of communication. No matter what tools and methods we use, if people are not willing to listen or to share their experiences and skills with others, the project will fail. Attitudes can be changed in a learning process - by thinking in different ways much can be gained.

Everybody seems to agree that the usability of a system or a product is essential, but somehow we do not seem to get there. When designing for a mass market, you have to design for an imaginary user - not real users. The resulting design solution does not meet the needs of the individual user; he or she must personalise it. On the contrary - when designing for specific users - you cannot solve the problems with generic packages. You may produce a perfectly good design, but people and requirements change. You have to maintain the design to meet the new requirements. Whatever the situation, a user-centred design approach is essential for the success of software development projects.

Thus, we argue that user centred design is far from being a lame duck. It is sorely needed, but it must be better adapted to the participants in the process and their needs. We therefore need to keep the discussion about UCD alive. We also need to do more research and field studies on the UCD process, particularly in the communication area.

References

[1] “User Centered Design – Problems and Possibilities” Jan Gulliksen, Ann Lantz and Inger Boivie (1999) *SIGCHI Bulletin, Vol. 31, No. 2, April 1999*, pp. 25-35. Summary of the PDC ’98 workshop on User Orientation in Practice – Problems and Possibilities. Also available with all accepted contributions as technical report TRITA-NA-D9813, CID-40. http://www.nada.kth.se/cid/pdf/cid_40.pdf

[2] International Organization for Standardization (1999) ISO 13407 (International standard) Human centred design process for interactive systems.

More Information

More information about the workshop, position papers from the attendants, as well as coming activities can be found through the first author or on http://www.nada.kth.se/cid/interact99/workshop/index.html

Acknowledgements

The workshop participants are acknowledged for their input to the workshop and for their input to this report. The workshop participants were:

Kirsten Bagger (Danfoss A/S, Denmark), Inger Boivie (TietoEnator AB, Sweden), Alan Doris (University of Edinburgh, U.K.), Sara Eriksén (University of Karlskrona/Ronneby, Sweden), Jan Gulliksen (Uppsala University & Royal Institute of Technology, Sweden), Susan Harker (Loughborough University, U.K.), Morten Brorup Harning (Copenhagen Business School, Denmark), Timo Jokela (University of Oulu &

Nokia Mobile Phones, Finland), Masood Masoodian (Odense University, Denmark), Rob Procter (University of Edinburgh, U.K.), Hans Erik Sørensen (University of Aarhus, Denmark), Harold Thimbleby (Middlesex University, U.K.), Nigel Claridge (Nomos Management AB, Sweden)

About the Organisers

CID is the Center for user oriented IT design at the Royal Institute of Technology in Stockholm, Sweden. This interdisciplinary competence centre combines user orientation and aesthetics in the design of usable, innovative IT. CID has a working group with the specific aim of promoting a user-oriented approach in all the research and development activities within the centre. One of the achievements of the group is USOR (http://www.nada.kth.se/~fredrikw/metod/index.html), a collection of user oriented methods. It also provides a forum for discussion in relation to user orientation.

Jan Gulliksen, Ph.D. in Systems Analysis performs research on user-centred design in several applied projects (e.g. with the Swedish National Tax Board) at Uppsala University. Jan is also supervising the working group on user orientation at CID.
Ann Lantz, Ph.D. in Cognitive Psychology, working with user oriented system design and communities at CID and a project on knowledge exchange, communication and context in electronic networks at IPLab, NADA, KTH.
Inger Boivie, MSc, works as a consultant with user-centred design and usability-related activities in a software engineering company. Inger also works part-time in the CID working group for user orientation.

Authors' Addresses

Jan Gulliksen
Dept. of HCI, Uppsala University
PO Box 337
SE-751 05 Uppsala
Sweden
email: Jan.Gulliksen@hci.uu.se
Tel: +46-18.471 2849

Ann Lantz
CID, NADA, KTH
Lindstedtsvägen 5
SE- 100 44 Stockholm
Sweden
email: alz@nada.kth.se
Tel: +46-8-790 68 17

Inger Boivie
TietoEnator AB
Kronborgsgränd 1

SE-164 87 Kista
Sweden
email: inger.boivie@tietoenator.com
Tel: +46-8-7036200

Easy-to-learn methods versus continuous learning in User Centred Design

Kirsten Bagger (bagger@danfoss.com) and Jacob Buur (buur@danfoss.com)
User Centred Design, Danfoss A/S
DK-6430 Nordborg, Denmark

Abstract

As a User Centred Design group placed centrally in a larger company, one is regularly confronted with the question of transferring competence to other functions within the organisation. These requests will be motivated by the need for expanding the activities or spreading the mind-set of user orientation. Frequently regular development and marketing staff move into usability related tasks and look for training.

So, it is not seldom that we as user centred design specialists are asked upon to convey our skills to others. An activity which is likely to fall in a dilemma between teaching 'easy-to-learn methods' (methods which are cooked up by others and a bit old - from when we did a presentation last) or a user centred design attitude (so that the learners can design their own methods through continuous experimenting and learning). The latter is definitely richer but much more difficult to convey. This is the dilemma we would like to address in our position paper.

Work practices are rapidly changing

User Centred Design is a very fast moving field, both technologically and in terms of work practices for design and user involvement. If we want to stay on the leading edge we need to adopt a practice of continuous experimentation and improvement of the way of involving users in product development.

The Danfoss User Centred Design group has worked with user involvement in product development for eight years. Looking back, our methodology has changed radically over the years: Starting from a mechanical design methodology basis [Buur et.al. 1991] with user interviews, via a cognitive engineering approach [Rasmussen et.al. 1994] favouring usability testing, to a participatory design philosophy [Kyng & Greenbaum 1991] with user involvement in design workshops [Binder ; Brandt & Horgen & Zack 1998] [Buur & Bagger 1999]. Along the way we have experimented with a range of methods from the HCI community to improve the usability of our products; use scenarios, drama, ethnographic field studies and design games among them. Some methods are now part of the work practice of the group, others vanished after a few years. Today the User Centred Design group at Danfoss is firmly embedded in (and an active developer of) the Scandinavian tradition; with the user considered an active participant throughout every product design process.

Projects shape mind-sets

That our methodology change continuously is not something we think much about. It does become very obvious, however, when we engage in projects with Danfoss divisions, which we have not worked with for a period. People who we collaborated with previously will confront us with the way we were thinking e.g. two years ago: Would you come and do a usability test for us, just like you did last time?

It is rather shocking to find out, that the way we work quickly establishes a mind-set in the organization: Status Quo thinking. This emphasizes the role of the user centred design specialists as responsible for continuous design learning in the organization.

With new collaboration partners we have found that there is no law of nature saying that we need to start up with the cognitive paradigm before we move on to 'true' user participation. 'Virgin' developers are quite susceptible to arguments from experienced user centred design specialists. It is when they have had their first experience with user involvement already, we find it difficult to move. So, there is no predestined learning path in User Centred Design, but if you stick to the methods you've always used you will get what you've always got.

An example: Panel of Experts (POE)

We will give an example of a method that was introduced at Danfoss four years ago. Panel of Experts (POE) is a method developed at the Design Technology Laboratory of Tektronix, USA around 1992 [Lynch & Stempski 1993]. It is a four-hour format for involving users in the early phases in product development. The workshop comprises a set of activities including questionnaire, dialog about work practice, a feature trade-off puzzle, prototype evaluation, and a brainstorming session. The method has suggestions for a preparation and evaluation process for the team too.

The method has a three-letter abbreviation and is a good handbook for involving users in product development. The engineers just love the quantitative part of the method and it has been slowly spreading in the organisation since a member of our group adopted the method during an intern at Tektronix in 1994.

However, the programme doesn't quite support the continuous, participatory relationship with users we prefer to establish today. We have used the POE methodology as inspiration for developing several types of user workshop formats and have testified the improvements through a number of projects. These new formats just don't have three-letter abbreviations, and we are reluctant to give them ones out of fear that they may turn into rigid methodology [Binder et.al. 1998].

Recently, we were approached by one of our product divisions to teach the POE method to their project managers. In itself this shows the keen interest in user issues, which is prevailing throughout our organisation: R&D staff is keen to include new working methods to improve the usability of products.

But how do we as user centred design specialists cope with this request? To introduce the methodology in its original form would take us one step backwards.

Teaching methods without rigor?

If we instead try to apply the user centred design methods creatively and to twist the methods so that we focus them on the concrete task at hand, there is a chance, that we can continuously improve our work practice. It seems that as soon as one writes up the methods in a practical form, the methodology freezes. The more effort you put into writing the book, the more resistant you become to change.

Doing User Centred Design by the book seems to lock the methodology and prevents you to try out new ways of bringing in the user's perspective and to improve your work.

We cannot claim that we have found a solution to the dilemma, but we do feel that this is an important issue to debate, if we want to prevent User Centred Design to become yet another 'here today gone tomorrow' methodology.

What we plan to do

We have decided to try out the following:

- Rather than introduce POE as *a* method, we will use a full case to give an overall view of a user centred development process. We will stress the collaborative activities and the learning nature of design. This places the POE method at a development level and shows user involvement throughout the development process.
- We will present the POE activity along with one or two alternative workshop formats and use this for discussing why different projects require different methods. We will stress basic user centred concepts and principles (tacit knowledge, reification, context, design iterations etc.).
- We will ask the participants to relate the cases directly to the projects they are currently involved with and help them characterise their own situation: Use context, users, organizational design context etc.
- We will then ask the participants to design their own user involvement process for their project, and use the variety presented to stress the point that a design process (method) is something you create specifically for your project and situation.

At the time of the workshop, we hope to be able to present our experiences.

References

BUUR, J., WINDUM, J., & JAKOBSEN, M.M. (1991) Man/Machine Interface Design Needs Systematic Methods. *Int. Conf. Engineering Design (ICED 91)*, Zürich.

RASMUSSEN, J., PEJTERSEN, A. & GOODSTEIN, L.P. (1994) *Cognitive Systems Engineering.* John Wiley & Sons, Inc.

KYNG, M. & GREENBAUM, J. (1991) *Design at work: Cooperative Design of Computer Systems.* Lawrence Erlbaum Associations.

BINDER, T., BRANDT, E., HORGEN, T. & ZACK, G. (1998) Staging Events of Collaborative Design and Learning. *5th ISPE Int. Conf. on Concurrent Engineering*, Tokyo, Japan.

BUUR, J. & BAGGER, K. (1999) Replacing Usability Testing with User Dialogue *Communications of the ACM, Vol. 42,* No. 5.

LYNCH, G. & STEMPSKI, M. (1993) User-Focused Engineering for Product Development. Tutorial notes. *Interchi '93,* Amsterdam.

BINDER, T. BRANDT, E. & BUUR, J. (1998) User-centeredness and product development Avoiding isolated UCD competency and the TLA trap, *PDC '98 Workshop.*

Intertwining Themes

Teaching Students How to Design New Technology for Actual Use and Bringing Design Issues to the Use Situation

Sara Eriksén (Sara.Eriksen@iar.hk-r.se)
Department of Human Work Science
University of Karlskrona/Ronneby, Sweden

Introduction

There are two main themes in this position paper. Both of them may be presented, most simply, as questions. One is about teaching and learning; How can Participatory Design – and User Centered Design – be taught?[i] The other is about focusing the use situation: How can we bring User Centered Design to the user, in the live use situation?

The challenge put forth to the participants of this workshop, how to make User Centered Design usable, may seem to come from a different direction. Here is a question posed from within the systems development process itself, where systems engineers are hounded by project managers and tight time schedules, and where UCD, at best, is represented and taken care of by a special facilitator. This is a work place where, traditionally, 'users' dare not tread. The workshop title question, however, is a profound one. It twists back upon itself. How do we make UCD usable – for whom? Put in this way, the usability issue becomes primarily about the systems developers themselves as users.

This puts old questions in a new perspective. How can we understand users and their needs? There are methods such as those used in ethnographic field studies, and there are, of course, approaches such as Participatory Design, and User Centered Design. If there is a third theme, then, one which is only just beginning to emerge, it might be presented thus; What are the work practices of systems developers? Are they, like technology itself, developing and changing? And, if so, what does this imply for UCD?

These are, to me, interesting and important questions concerning the areas which I understand your workshop will be exploring, above all through the sharing of experiences from various case studies.

Seeing by juxtapositioning; User Centered Design versus Participatory Design

While reading through the report from the workshop *User Centered Design - Problems and Possibilities*[ii], I came across the distinction made there between Participatory Design

[i] Compare Helgeson, Sutter and Eriksén, 1996, *How can Participatory Design Be Taught?*
[ii] Held at PDC'98 in Seattle, November 14th 1998, see Gulliksen, Lantz and Boivie, 1998.

and User Centered Design, and the discussion about to what extent they may, and may not, overlap[iii]. Having mainly worked with methods and ideas stemming from a Participatory Design approach, this was something I had not really thought much about previously. To me, it seemed provocative and paradoxical that User Centered Design could call itself by this name, even when users were not directly involved in the design process[iv]. This juxtapositioning of concepts, however, helped me to understand more about the complexities of the problem areas which they both bring into focus, albeit from slightly different perspectives. One of the issues which was agreed upon by the workshop participants in 1998 as being a real and immediate challenge to Participatory Design, was to ensure that user participation doesn't become ***only*** that. User Centered Design, by its very name, signifies that the designers center their designs on the user and the users' needs.

This problematizing of concepts lead me to reflect once again on some of the experiences I have had during the past few years with systems development projects where users have been involved. I am at present planning yet another research project of this type. It is a project where the medium is the Internet and the users are the general public, an incredibly vast and heterogeneous 'group', yet one being more and more commonly addressed by new applications of information and communication technology these days. When the user could be, literally, almost anyone, almost anywhere, almost anytime, it becomes necessary to begin by trying to visualize 'most likely use situations'[v]. Back to square one? Not quite. More like starting with a prototype, and thus bringing design out into the open, in to the actual use situation. In this light, iterations of design solutions, and multi-disciplinary design teams, two of the basic principles of User Centered Design[vi], become a life-cycle issue and a concern for everyday IT management.

More and more, the applications being developed are based on off-the-shelf, but basically generic, products, which need to be further designed and developed in use in order to become usable and useful. Systems development projects are challenged with, on the one hand, compatibility and standardization issues over which they have little or no control, on the other hand the need to develop flexible, reliable and usable applications for real use situations. It may be that not only methods and techniques need to be refined to meet these challenges, but that new ways of thinking about systems development projects in trans-organizational contexts are called for. These seem actually already to be emerging in for instance the changing work practices of computer consultant companies, where services around up-grading, customizing, help-desk functions, personnel education etc. are becoming as important as the development of new software applications. Juxtapositioning User Centered Design and Participatory Design may be one way of theoretically reframing and gaining a better understanding of systems development work practices and how they can be supported, by methods, techniques, IT and the surrounding organization(s).

[iii] Gulliksen, Lantz and Boivie, 1998, p. 8-9.
[iv] Ibid., p.8. Although one of the four basic principles for User Centered Design listed in the report is 'active involvement of users', there were cases presented at the workshop which were classified as user-centered, but which did not in any true sense have user participation in the design process.
[v] Eriksén, 1999.
[vi] Gulliksen, Lantz and Boivie, 1998, p.8.

How can Participatory Design – and User Centered Design – be taught?

Besides doing research, I spend half my time teaching within a Masters' program called *People, Computers and Work*[vii], where we combine equal amounts of Human Work Science and Computer Science in educating systems developers for the future. In teaching, we focus very much on ***IT in use***, on letting the students study actual work practices and real life problems as a basis for understanding the contexts within which they will be designing and developing IT solutions. Therefore, the question of what systems developers actually do to get their work done, i.e. what their work practices are, is relevant for us in at least two important ways[viii]. First of all, systems developers, like other users, need computer support which supports the work they do and helps them get it done as smoothly and effectively as possible. Secondly, how can we teach our students what they need to know, if we don't know and understand something about the kind of work they will be doing in the future? I find examples from case studies to be very useful in teaching. I use them as a way of helping students envision what systems development is about, and what kinds of decisions and actions it involves, not only on an abstract, strategic level, but in the everyday activities which, finally, determine what gets done and what doesn't.

In the autumn of 1999, the third-year students of the *People, Computers and Work* program will be taking three courses, which will to some extent run in parallel. The courses concern Human Computer Interaction (5 points[ix]), Participatory Design (5 points) and IT Design (10 points). In previous years, these courses have each been held independently. Three different teachers have been in charge of them, and they have had little or no contact with each other in their planning and teaching of the courses. This year, we have decided to deliberately make all three courses interdependent on each other, by coordinating our themes and carrying discussions, examples and project work through from one course to another.

IT Design is a course in which we strive to move beyond the desktop metaphor and take a closer look at design concepts and ideas concerning ubiquitous computing, tangible bits, mobile computing, etc. One of the basic questions addressed in the course is; What is design quality? This is, of course, a basic question for both Human Computer Interaction and Participatory Design as well. By beginning to coordinate the courses, we hope to integrate issues which we feel should be naturally integrated in real life systems development. We are expecting an initial period of chaos. However, we are striving to develop an atmosphere which acknowledges real world ambiguity and diversity, and which encourages productive problematizing and creative use of methods for managing complexity in systems development.

[vii] The Swedish acronym for this is MDA, which stands for 'Människor, Datateknik och Arbetsliv'.

[viii] Two of my colleagues at the University of Karlskrona/Ronneby, Dr. Yvonne Dittrich and Gunnel Andersdotter, are at present involved in a research project where ethnographic fieldmethods are being used to study work practices in large software development projects.

[ix] 1 point in the Swedish academic study credit system equals approx. 1 week of studies/project work.

How can we bring User Centered Design to the user, in the live use situation?

Since 1992, I have been involved in research projects focusing on the use, design and continual support and development of computer support for public administration in one-stop shops[x]. During the past few years, with the rapid expansion of the use of Internet/intranet solutions within public administration, this has come to encompass the on-going integration of such basically traditional administrative support systems with public electronic information systems. This has, in turn, lead to the realization that new information technology is developing faster than the models, metaphors and methods used for conceptualizing the sharing and managing of information in organizations, in communities and in society in general.

The way we utilize information technology today does not seem to succeed in supporting the everyday work practices through which organizations accomplish their work. It seems we need not only to be aware of multi-perspectivity as an issue, but that we should make better use of it in the design and continual support of information systems in use. Many methods for systems development are designed to diminish rather than make use of ambiguity and diversity. And even methods which encourage multi-perspectivity, such as UCD, may be forced into defensive positions in the systems development context because of organizational limitations, the way our roles are defined and delimited, and because of assumptions about how work actually gets done which are built in to the tools we apply.

What I found, in my case studies, was that much of the work which was getting done at the front desks of one-stop shops was accomplished through the skillful managing of ***complex situations***. Whereas the computer support was obviously designed for managing one stand-alone task at a time, usually in a given sequence of steps which had to be followed in order to accomplish the task at all. There was little or no support for desk-level intentional action, i.e. tools for planning and following through of various one-person or work-group level projects, no communication and feed-back processes besides e-mail, no thought-through support for managing interruptions and moving back and forth between different applications. There seemed to have been no awareness at all about the actual use situation in the design of computer support for front office work.

The question I finally found myself formulating was; How can ***IT management***, understood here as concrete, visible and locally accountable design and management of IT support for everyday work practices, be made visible, be supported and be continually designed and developed in everyday use as an asset for the informating[xi] organization? How can we bring design issues to the actual use situation, and build them in to the working concept of what managing organizations and IT is all about? That, to me, is an important issue for UCD.

---

[x] In Sweden, these one-stop shops are most often called *Medborgarkontor*, which, directly translated, would be Citizens' Offices. See Eriksén, 1998.
[xi] Zuboff, 1985, makes the distinction between automating and informating in her article Automate/Informate: The Two Faces of Intelligent Technology.

Bibliography and background

I am at present a lecturer in the department of Human Work Science, at the University of Karlskrona/Ronneby in southern Sweden. Here, I teach within an interdisciplinary Masters program called *People, Computers and Work* (*MDA* is the Swedish acronym), combining Computer Science and Human Work Science in educating systems developers for the future. I am currently involved in a research project focusing on the use, design and continual support and development of computer support for public administration in one-stop shops, and the on-going integration of such systems with public electronic information systems. This is a research project which is being financed by the Swedish Council for Work Life Research[xii]. During the past three years, I have also participated in the EC project ATTACH (Advanced Trans-European Telematics Applications for Community Help, UR 1001, 1996-98), in which the University of Karlskrona/Ronneby was a partner. I am at present involved in the UK-/Nordic Initiative on Information and Communication Technologies (ICTs), which is being sponsored by the Research Councils of Sweden, Finland, Denmark, Norway and the UK[xiii].

The University of Karlskrona/Ronneby was founded in 1989. It is a young and small, but rapidly expanding university, with approx. 3,000 students and 330 employees. The main emphasis in both research and teaching is on ***IT in use*** – i.e. on information technology and how it is used. In teaching, we emphasize problem-based learning. Students work in projects, often in collaboration with businesses and other organizations. Cross-disciplinary course modules and cooperative projects are offered, involving students and staff from different subject areas. Networks and contact points between the University and regional industries, small and medium-sized enterprises and other organizations are also deliberately cultivated and supported. Through research projects as well as student projects, through organized meetings and visits, theories can be put to test; thus they are put into 'real life' proportion by being presented in the context of, and directly related to, everyday worklife experiences and needs.

References

ERIKSÉN, SARA (1998), *Knowing and the Art of IT Management. An inquiry into work practices in one-stop shops*. Ph.D. thesis, Lund, Sweden: Department of Informatics, Lund University.

ERIKSÉN, SARA (1999), *Globalization and 'Genius Loci'; From Intentional Spaces to Purposeful Places.* Working paper for UK-/Nordic workshop held in Ronneby April 15th-16th 1999 on the theme 'Social Theory and ICT'. Website http://www.brunel.ac.uk/research/virtsoc/nordic/eriksen.htm (99-07-15).

[xii] No. 97-0242. The Swedish Council for Work Life Research also financed the research project which preceded this one, no. 94-0349, and which focused on skill, cooperation and computer support in public service front offices.
[xiii] See Eriksén, 1999.

GULLIKSEN, JAN, ANN LANTZ & INGER BOIVIE (1998), *User Centered Design in Practice – Problems and Possibilities*. CID report TRITA-NA-D9813, CID-40, ISSN 1403-073X, also available from the website http://www.nada.kth.se/cid/pdc98/workshop (99-06-05). Report from workshop held at PDC'98 in Seattle, USA, November 14th 1998.

HELGESON, BO, BERTHEL SUTTER & SARA ERIKSÉN (1996), *How Can Participatory Design Be Taught?* Paper and workshop held at PDC'96 in Cambridge, MA, USA, November 15th 1996.

ZUBOFF, SHOSHANA (1985), Automate/Informate: The Two Faces of Intelligent Technology. In *Organizational Dynamics* Autumn 1985, pp. 4-18.

Organisation and Communication for Usable UCD.

Jan Gulliksen[1,2] (Jan.Gulliksen@hci.uu.se), Inger Boivie[2,3] (inger.boivie@enator.se) and Ann Lantz[2,4] (alz@nada.kth.se)
[1] Department of Human Computer Interaction (HCI), Information Technology, Uppsala University, Lägerhyddsvägen 18, SE-752 37 Uppsala, Sweden
[2] Centre for user oriented IT design (CID), Royal Institute of Technology (KTH), Lindstedtsvägen 5, SE-100 44 Stockholm, Sweden
[3] Enator AB, Kronoborgsgränd 1, SE-164 87 Kista, Sweden
[4] IPLab, NADA, Royal Institute of Technology (KTH), Lindstedtsvägen 3, SE-100 44 Stockholm, Sweden

Abstract

With a user centred design process we presume to end up with a usable product. Unfortunately, very often, this is not the case. The many reasons for this have been outlined in several publications, e.g. attitudes, methods and tools, time, competence, commitment and management support. This position paper mainly focuses on issues relating to communication and to project and work organisation since these are key factors without which no usability related efforts will work. It discusses the role of the creative designer and its impact on usable systems. It describes some preliminary results of studies of how GUI designers work and how this knowledge can be used to influence project organisation and work.

Introduction

Undoubtedly, very few would question the relationship between usability and a user centred approach. Active user influence is essential for the design of usable systems. How come, then, that the users often are left out or considered an aggravation in development work? Is it a matter of attitude, maturity or management? Is it a matter of project organisation or is it just another example of how difficult it can be for people to communicate and respect each other? We guess the answer is: “Yes, it is all of these!”

According to Gould and Lewis user interface developers often tend to believe that good interface design is a matter of getting it right the first time [Gould & Lewis, 1985]. However, usability requires continuous iterations in which real users are given the opportunity to evaluate the system and its interface. A user centred approach to systems development and design should in all situations be preferred. The main reasons being:

- The end users are experts on their tasks and therefore the only ones that can describe it properly.
- The end users are suitable for testing and evaluating prototypes and systems that are developed for them.

But, on the other hand, user participation in a development project is never, in itself, a guarantee for a usable system. Abundant evidence of this is furnished by the large number of computer systems with severe usability problems that exist in working life today and the vast number of projects that have failed. Via interviews with representatives working with usability issues on the field, and our own experiences of usability work, we have identified and further analysed the following user orientation problem areas:

Problem Areas

Even when some of the critical success factors are in place, such as having **management support**, the participants **commitment** and actually performing design according to an **iterative** process, user centred design is not a straight-forward road to success. Other problem areas detected are:

- **Attitudes** - System developers do not regard themselves as service providers, rather many system developers regard computer systems development as a creative occupation or a plain problem solution activity (i.e. technical problems, rather than problems within a work context)
- **Methods and Tools** - Methods and tools exist, but are they useful and easily available? Some methods are only available as standards or commercial methods and therefore not accessible to the public.
- **Time** - An iterative development process is a pre-requisite for user centred design [Gould & Lewis, 1985]. But there is rarely enough time for iterations in development projects. The construction phase tends to delay the project. When the time comes to perform a usability evaluation, it is not uncommon that time has run out - leaving room for a final evaluation only, resulting in a usability assessment but no significant changes to the system.
- **Competence** - Competence is a concept in itself problematic. In the Webster's new abridged Oxford edition competence is defined as having the suitable skill for a specific purpose. We rather view competence as the ability to handle and manage the situation in a specific work context. Competence includes knowledge and experience of the work activity, and social skills, such as the ability to co-operate and communicate in a group. The participants in the design process rarely have the knowledge, competence, special abilities or even interest in working in a user centred fashion. HCI is, despite decades of research, still relatively unknown to most participants in practical system development. Moreover, the results of the HCI research are often difficult to communicate and difficult to apply.

Communication

We do, however, believe that communication is one of the most crucial factors for success in designing usable systems**,** all other problem areas being inextricably connected to the communication abilities of the project members. Saarinen and Määkinen [1990] has pointed out that the success of a systems development project is little related to the amount of user involvement as measured in the time spent by the users in the project and the number of users participating. Essential, though, is the quality of the user involvement, and above all the ***systems*** developers' communication skills.

Many roles are involved in a ***systems*** development project and in this position paper we focus on the role (and attitudes) of the systems developer. As for successful communication with the users we believe that the responsibility rests with the systems developers**.** It is essential that the systems developers understand the context of use, the users and their tasks. Thus, it is important that the systems developers make an effort to understand the terminology used by the users to describe their tasks. Moreover, the systems developers need to phrase their questions so that they get the information required, without intimidating the users with too much technical jargon.

But, what about the other people in the project? What about graphic designers, HCI experts, etc? Does the responsibility for successful communication rest with the systems developers only?

For instance, how come the methods used within the HCI field sometimes fail to improve the usability of the system? Is it simply so that the results of, for instance, a usability evaluation are not communicated in a way that is suitable for the systems developers? Do scenarios, task descriptions, user profiles, etc, provide the information the systems developers need to create the system? Are the HCI methods adapted to the reality of a systems development project? Design involves making decisions on all levels from deciding on a suitable conceptual model to deciding on the design of individual elements in the user interface. The systems developers often need quick answers regarding such design matters, or they will not meet their deadlines. A user study or a perusal of the Windows Guidelines, may not be what the developer had in mind, asking the question. If the systems developers do not get the answers they need, they will stop asking questions. If the only thing they get out of the HCI expert is criticism of their design at too late a stage, they will stop listen to him/her.

How do graphic designers or industrial designers communicate with the systems developers? Sketches, story-boards and prototypes are great ways of communicating ideas, but what if the ideas turn out to be impossible to implement given the technical constraints of the project? At the PDC workshop [Gulliksen, Lantz & Boivie, 1999] one of the participants told a true story about the industrial designers, who went off on their own to ”create” the design. They were not interested in working together with the users or any other project members. The ideal being the lonely artist coming up with brilliant ideas in his/her solitude. Does not such behaviour imply that the designer considers him-/herself a little bit above the other people in the project? They are there only to make the designer's brilliant ideas come real. As Donald Norman stresses from time to time: Designers do never explicitly have the goal of designing usable systems, designers aim for design prices, and in rare cases, by coincidence, the design turns out to be usable [Norman, 1998].

Organisation

Several of the matters raised in relation to communication of course also relate to the organisation of the development project and the organisation for which the new system or technology is being produced. Donald Norman stressed the impact the HCI community can have in the development of usable products by explaining the need for usability professionals acting as peers [Norman, 1998]. Such usability professionals should work in close co-operation with the marketing department rather than relying on the individual project manager to be interested in including usability activities in each project. Conversely, system developers should not be regarded as peers, a role they often shoulder as a consequence of the unique knowledge they posses. Rather, they should be regarded as technical expertise, implementing the design solutions that have been created in a user centred, iterative, fashion in close co-operation between users, design expertise and the system developers.

In a recent, still not published, study of the way GUI designers actually work in a large Swedish government organisation it is obvious that system developers do not, and do seldom want to design. The actual design of the user interface is something that occurs without any specific design effort. Design is not a creative process, it is an engineering task, that produces formal models rather than systems that are adapted to be used by users. Similarly, the task of the GUI designer is to solve problems of technical nature, such as writing the program code for a specific subset of the system - a successful result being that the code complies with performance requirements and does not contain bugs. When the system is shown to the users, they complain about “irrelevant details” and do not see the smart coding procedure.

To make projects work and produce systems of an acceptable level of usability within the limited time frame the following guidelines need to be addressed:

- **Project size** - Do not make the projects too large, and if large projects are necessary, make sure to produce deliverables frequently, e.g. at least once a month.
- **Speed up the initial phases** - Try to get to the user interface design process as *quickly* as possible. There is no need to finish an analysis phase and have fixed requirements before starting to design.
- **Clear goals** - Make sure that the goals of the system and the supported work activity are clear at a very early stage, so that scenarios of the future work activities can be produced as a means for starting the process of conceptualising the problem.

It is only by making the user centred design process more usable for the involved parties that we are able to achieve the goals above.

Conclusions

Management support, commitment, a user centred design process and appropriate methods do not suffice to guarantee a usable system. The crucial factor is the ability to communicate within the project and with the users. Given that the responsibility for successful communication with the users rests with the system developers, how do we improve communication within the project? I.e.

- How should the HCI communicate the results of user centred activities and usability activities?
- How should designers communicate their ideas - and win the support of the system developers for those ideas?
- How do we create an organisational context and culture that promotes rather than prevents usable user centred design?

- How do we make the user centred design process usable for its users?

References

GOULD, J.D., & LEWIS, C. (1985) Designing for Usability: Key Principles and What Designers Think. *Communications of the ACM, Vol. 38, No. 3,* pp. 300-311

INTERNATIONAL ORGANISATION FOR STANDARDISATION (1998) *ISO/IS 9241. Ergonomic Requirements for Office Work with Visual Display Terminals (VDTs)* Part 11: Guidance on Usability, International Standard.

SAARINEN, T. & SÄÄKSJÄRVI, M. (1990). The missing concept of user participation. An empirical assessment of user participaation and information system success. In *Scandinavian Journal of Information Systems, Vol. 2,* pp.25-42.

GULLIKSEN, J., LANTZ, A. & BOIVIE, I. (1999) *User Centered Design in Practice - Problems and Possibilities.* Technical report number: TRITA-NA-D9813, Available: http://www.nada.kth.se/cid/pdf/cid_40.pdf

NORMAN, D.A. (1998) *The invisible computer.* Why good products can fail, the personal computer is so complex, and information appliances are the solution. The MIT Press, Cambridge, Mass.

Delivering user-centred design tools to project teams with no usability specialists

Susan Harker
Department of Human Sciences
Loughborough University
Loughborough Leics LE11 3TU UK

Over the past fifteen years I have been one of a group of people, based in the Department of Human Sciences and the HUSAT Research Institute, engaged in delivering tools to promote the application of user-centred approaches by members of IT project teams. Two basic assumptions have underpinned these activities,

1) that there is a need to increase the level of user-centredness across a broad spectrum of design processes and domains, and
2) that it is unrealistic to expect that there will enough specialists in UCD to undertake all the work required to bring about the necessary increase in the level of user-centredness.

The initial focus of the work was to develop the tools themselves. This process itself was one which had to be user-centred. Members of design teams from a variety of disciplines present different requirements from those of the experienced human factors practitioner. The nature of the design task also has to be understood and taken into account. Perhaps the most difficult challenge is to carry out the necessary evaluation of the tools as they develop. In developing the ORDIT toolset for example we had to progress from tests of elements of method, through simulations of the application of the whole method, back to testing parts in real settings and ending with partial tests of the whole method on actual design tasks. The ultimate success of the method can only be judged when (and if) it passes into more general use, without support from the members of the research team.

It is perhaps not surprising that, even though this work started in the middle of the 1980's, lessons have continued to emerge up to the present time. One reason for this is that initial efforts were intentionally directed at the development of a generic methodology which was modular in form. The expectation was that it would be necessary to carry out local modifications to suit the different design environments in which it was to be used, both in terms of the application domain and existing design practices. Experience has confirmed this. Even in situations where organisations start with the view that the existing tools will slot directly into their environment, it has proved necessary to make various changes to accommodate local circumstances. For example in translating the tools for use in a military project setting, there needed to be far greater emphasis on the analysis of the stress factors which the task environment imposed on the end users. In another case, where the aim was to support systems integration activities in manufacturing environments, it was necessary to pay more attention to the use of the tools within the framework of a multi-disciplinary design team representing the interests of different companies contributing to the integrated solution.

With the growth in experience of how to successfully disseminate the use of the tools, other issues arise. Somewhat to our surprise, given that the aim of the majority of the tools has been to support non-human factors specialists, their most enthusiastic users have often been the human factors practitioners. There are a number of conclusions which may be drawn from this. The first relates to the assumption that we can deliver user-centred tools in a usable form. Early versions were typically complex (not that this was what we intended at the time!) and relied on levels of knowledge and skill which only the human factors people had. With iterative evaluation the need to simplify and clarify what has to be done becomes evident.

Another factor which has grown in importance is that of training. It was always assumed that the tools had to be delivered in conjunction with introductory training courses and practical exercises. What has emerged as an additional requirement is the need to follow up the introductory sessions with opportunities to reflect on experience in practice and to exchange ideas with others. A related issue concerns the process of institutionalising the use of user-centred processes and the associated tools. One thing which emerges from the feedback from those people who have participated in training is the difficulty of being a lone voice in promoting such an approach. Even when there is an organisational commitment to being more user centred, a single representative on a project team charged with pursuing the user-centred agenda finds it almost impossible to carry the rest of the project team along. A significant proportion of the team need to understand and be in a position to contribute to the use of the tools.

These topics take us into the realm of gaining organisational commitment and changing the design culture. Clearly tools do not and should not be expected to serve this function. They do provide the advantage that they offer tangible evidence of systematic support for such moves.

One thing which should be addressed before concluding this brief account is the question of the risks associated with handing over responsibility for including user-centred activities in the design process to those without specialist skills. As with all simplifications there are circumstances where the work requires specialist input and more detailed attention to the problematic areas of the user-system interaction. There are open questions about how to ensure that these cases are identified and dealt with.

Making UCD possible as part of a structured design method

Morten Borup Harning (harning@cbs.dk)
Department of Informatics, Copenhagen Business School
Howitzvej 60, DK-2000 Frederiksberg, Denmark
http://www.cbs.dk/~harning

Background

Usually UCD and structured design method are opposing ways of approaching user interface design. However it does not need to be that way. During the last several years I have been working on a structured design method that allowed the designer to structure the design process, with the advantages this can have when addressing complex user interface design problems, while making it possible to evaluate and discuss different designs with potential users.

The structured design process

My work has been focusing on the design process of mapping different types of requirements into a concrete user interface design. I propose to structure the design process into the following four design phases [Harning, 1996; Harning, 1997]:

1. Conceptual design
2. Logical design
3. Dialogue design
4. Physical design

The conceptual design phase focuses as the name suggest on the conceptual design, identifying the necessary concepts and associated information based on task requirements. Often the list of task will be available however if this is not the case a list of task will need to be compiled. This can be done in parallel with conceptual design or as an iterative process where the focus changes between designing concepts and describing tasks.
The result of the conceptual design is a so-called user data model that consists of a form per concept. The form contains a label or some other kind of visual identifier and an exemplary value for each of the attributes associated with the concepts. The user data model form will also include visual cues describing the relationships between the concept described in the form and other concepts. The design guidelines for designing such user data models can be found in Harning [1996]. The conceptual model will typically also be described in a more formal way e.g. as an entity-relationship model.

The conceptual design focuses on identifying the necessary set of concepts that will allow the user to *model the context* in a way that is *appropriate with respect to the task descriptions*.

The logical design is in many ways closely related to the conceptual design, however the focus is on making sure the design satisfies the information demand of each task. The conceptual design ensures that all information described as concepts are available, but does however not address task efficiency. The goal of the logical design is to ensure a reasonable level of task efficiency while maintaining a conceptually clear design. The logical design is divided into two closely related parts, one focusing on the visual design, designing so-called logical or virtual windows, and one focusing on the functionality needed seen from a logical point-of-view, designing so-called user functions. User functions can be thought of as high level "buttons" that needs to be available in order to perform the data manipulations required in order for the user to perform the tasks described. A detailed description of the visual part of the logical design (that is the design of the logical windows) can again be found in Harning [1996], whereas the process of designing user functions can be found in Harning [1997] or in a earlier version in Lauesen & Harning [1993].

The third phase, called dialogue design, aims at designing and describing the detailed user interface dialogue, e.g. the flow between windows and identifying necessary states or modes describing the availability of user functions in different parts of the user interface. This ends up being a very detailed description of how the user interface should be implemented. This part of the design process has only been described in Harning [1997], but is in many ways related the dialogue element found in many other user interface design methods.

The fourth and final phase is concerned with fine-tuning of the design, e.g. designing button placement, combining logical windows into clearly understandable physical windows, describing how error situations should be handled including the form of error messages.

Why is the structured design process important?

The need for structured design method become apparent when the design space grows, as is often the case when designing complex user interfaces. One of the problems with typical iterative design methods is that it is difficult to decide when no more iteration will be needed. Another problem is that main ideas of the design is often lost from between too iterations (especially of the iterative design process aims at exploring to complete design space) or when implementing the final prototype. These problems can be addressed by a structured design method. By structuring the design process into smaller parts the method makes it possible for the designer to ensure that, all major aspects of the design has been addressed. In the method that I propose the completeness of the conceptual model can be tested by checking that the information needed by all of the identified task can be found in the conceptual model. With respect to the problem of loosing design ideas this is addressed in a structured method be describing the relationship between different design parts.

Why is UCD typically a problem in structured design methods?

The notations used in most structured design methods are typically rather abstract and hence difficult to understand for the unskilled designer. If we try to let the user participate in the design process, e.g. by asking the user to evaluate or comment the design ideas, the result of such an evaluation will only be of value if the user is capable of fully understanding different aspects of the often detailed notation. These problems make it difficult for the user and designer to communicate about possible design problems.

The visual design products seems to be the key to the solution

Structuring the visual design process as described in this paper (and in more detail in Harning [1996]) makes it possible for the user to relate the crucial parts of the design. Especially the user data model, the logical windows and finally the physical windows, that from a UCD point-of-view are very similar to a tradition prototype, seems to enable practices typically only found in more traditional UCD approaches.

Experience so far

The structured design process described here has been used in several large design projects. It has been used for designing of a classroom scheduling system that has been in use for the last several years managing all classrooms at Copenhagen Business Scholl. It has been used for designing web-based system used to organise lecture materials, lecture plans, course descriptions and messages supporting a 3 year bachelor programme, and 2-3 year master level program. Most recently it has been used for designing the instrument panel for a novel kind of vehicle combining the qualities of the traditional car and the traditional train.

Important for all of these projects has been the possibility to validate first the conceptual design and later the logical design before moving on in the design process. I all cases a user centred design has been important. The experience so far has been that especially the visual design products such as the user data model, the logical windows has made it possible to get crucial user feedback even when designing rather complex systems.

In the design of web-based system it proved to be crucial however, that only the parts of the design that the user would later expect to be able to edit was shown as editable information. Hence the conceptual design had to be presented differently depending on the user group. Seen from UCD perspective this is however a natural extension to the design method.

Literature

HARNING, M.B. (1996) An Approach to Structured Display Design - Coping with Conceptual Complexity. In J. Vanderdonckt (Ed.) *Computer-Aided Design of User Interfaces*, proceedings of the 2nd International Workshop on Computer-Aided Design of User Interfaces CADUI'96 (Namur, 5-7 June 1996), Presses Universitaires de Namur, Namur, pp. 121-138. ISBN 2-87037-232-9.

HARNING, M.B. (1997) *Software User Interface Engineering (SuiE) - a structured method for designing user interfaces* (in Danish), , Ph.D. dissertation, Faculty of Economics, Copenhagen Business School, Ph.D.-series 5.97.

LAUESEN, S., & HARNING, M.B. (1993) Dialogue Design through Modified Dataflow and Data Modelling. In T. Grechenig and M. Tscheligi (eds) *VCHCI'93 proceedings*, Vienna, September, 1993, pp. 172-183.

Making User Centered Design Usable to Managers

Timo Jokela (Timo.Jokela@oulu.fi)
University of Oulu & Nokia Mobile Phones

Background

How to introduce user centered design smoothly and effectively into a product development organisation is a recognized problem in usability engineering community. It is a topic of many presentations and panels in usability conferences and seminars (for example, there were tutorials, panels and interviews on the issue at CHI99). I think that many practioners – possibly also academics - face this issue as a problem in our daily work. The problem is pragmatically relevant. Some organisations have just recently taken their first steps in user centered design, and many others are only at a stage where they have picked up the idea that they should do something in this area.

There are a number of articles in magazines that present ideas for how to introduce usability. A typical approach is to select an appropriate project and start doing some usability activities – typically usability tests while it is a concrete and easily understandable activity [Aucella, 1997; Fellenz, 1997; Tudor, 1998]. While this kind of bottom-up approach might work well in short term, there may be pitfalls in long term. For example, a widely known experience is that when the schedules of projects are tight, usability work may be skipped because 'we don't have time to it in this project'. Another typical problem is that usability persons are not involved at higher level decision making. Decisions about essential usability related product properties are made without usability persons participating in the discussions. Usability testing – we must admit – is often regarded only as relatively 'low level work' [Norman, 1999].

One major reason for this kind of problematic status of usability is that management does not have a clear understanding about the essentials of usability and user centered design. Management may have lack of understanding what actually usability is, what are the benefits that are gained from user centered design, and what is the impact of user centered design to the development process.

Another reason may be that the role of usability is not that important in all businesses – and we usability professionals should be humble enough to admit that. It is quite natural to think that usability is not equally important in all organizations but is dependent on the product and the business of the organization. In some cases, usability may be one of the most critical quality features while in some cases many other product characteristics may be bigger drivers [Browne, 1998]. This is typically the case with with those products products that are aimed for early adopters [Norman, 1999]. The number of features and design – those product characterstics that are important in purchase situations - may be clearly more important than usability that typically can be experienced only after purchaing situation, in the daily use of a product.

Our basics

The approach that we present in this position paper represents a very top-down paradigm of how to introduce user centred design – or how to 'make user centered design usable' - in an organization. The basic message of this position paper is that one potentially effective approach to start introduction by analyzing the potential *business benefits* that usability would bring to the organization. *Business benefits make user centered design usable to management.*

After the business potential is explored, the role of usability in the strategy of the organization is defined. Naturally, the bigger the potential business benefits are, the bigger role user centered design may have in the strategy. And only after usability is planned at the strategic level (and thereby management commitment to usability gained), the introduction of user centered design at pragamtic level is started. Figure 1.

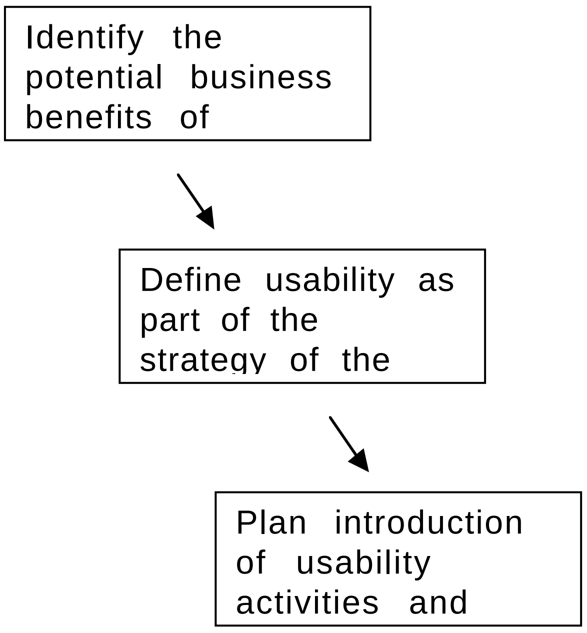

Identify the potential business benefits of
Define usability as part of the strategy of the
Plan introduction of usability activities and

Figure 1. First steps of introducing usability

With this kind of approach, the introduction of pragmatic level usability activities – e.g. usability testing – is not started immediately. However, we hope to get a firm basis for long term development of the organization towards the culture of user centered design.

On the other hand, one should understand the potential risk that this kind of approach would mean to usability persons. The analysis should be done as objectively as possible. The result may be that usability does not give very important benefits in that business. Naturally, this means less investment in usability work, probably also less status of usability group in relation to some other groups. But probably it is good to face this fact at the early point rather than get frustrated in long term. (However, we believe that this kind of scenario is rather seldom the case.)

Anyway, we hope that the pragmatic implementation of user centered design activities could be done smoothly and successfully with this kind of approach (even if we do not have data on this yet, see section Status of work). We also hope that this kind of approach would raise the status of usability personnel and respect of usability work.

A top-down oriented approach has been presented e.g. by [Bloomer & Croft, 1997] and [Billingsley, ?] - even if they do not talk about 'top-down'. So, our approach is related to theirs. Our strategy is to put even more focus on the systematic business benefit analysis. We try to avoid the spirit of selling usability; instead we wish to be able to do the analysis objectively. In addition, while they have addressed in-house systems development, we address vendor companies. Therefore, we don not talk only about 'cost-saving' but also competitive advantage.

Pragmatics

We explore here a bit more detailed the first step: how to analyse the usability business benefits. Our basic approach is to *interview all the key managers* of the business in focus. (While we are mainly focusing on small and medium size enterprises, having interview sessions with key managers should be achievable).

Before the interviews, it may be useful to get a rough idea of the areas where the main potential usability benefits may be. A cheklist for such 'home work' might include following kind of points:

- Is productivity a key issue in the end user organization?
- Are there critical tasks that are done with the product?
- Is the end user population large?
- Are the support costs significant?
- Are the training costs significant?
- What is the usability status of competitor products?
- Is the product used for accessing chargeable services?
- Is the product aimed for public use?

In the interviews, we try to explore what kind of benefits potentially could be gained with user centred design in each specific function (marketing, development, customer support etc.), as well as the organization as a whole. In the discussion, we use two categories of benefits

- Cost-saving: what kind of savings improved usability of products would bring to the organization
- Competitive advantage: what kind of sales arguments would usability bring. These arguments should naturally arise from the business benefits that improved usability of products would give in the customer organizations.

We hope to be able to get not only qualitative but also quantitative data (working hours, number of help desk calls etc.) about the areas of potential benefits. If they do not have measured data – as typically is the case in small companies – we hope to get at least estimates. We also wish prioritize the different benefits, to identify the most important ones.

One significant viewpoint is that these kind of interviews increase the management awareness about usability. In order to make discussions of the potential benefits successful, the essentials of usability needs to be introduced in the beginning of an interview session. On the other hand, this is a challenge of its own.

The overall vision is to gain a situation where we can hear management representatives saying that 'We do user centered design in order to achieve remarkabe savings in help desk services', or 'We do user centered design in order to have the acceptance from public users', or 'We do user centered design in order to gain competitive advantage through helping our customers to sell more services'.

The step from the strategic planning of usability to the planning and implementation of concrete usability activities is still open. We hope that we could develop a path from strategic usability needs to appropriate user centered design activities.

Status of work

This work is being done in a national research project 'Käypro' (Enhancing User Centredness in Product Development Processes) in Finland. The early part of the project focused on the assessment of the usability maturity of development organizations and projects. For example, we experimented [Kuutti et al. 1998] the maturity models developed by the INUSE project [Earthy, 1998]. Now we are at the stage where our focus is make the improvement happen.

In the context of the Käypro project, we have agreed to do usability business benefit analysis as described in this paper in some companies. The work has just recently started: we did the first interview only recently. By the time of the workshop at Interact'99 workhop we should have done probably about ten interviews at two companies.

Our guess is that the contents of the interviews will be refined case by case. (Designing a good interview is iterative work as user centered design is…) We hope to get an understanding how to conduct management interview in an appropriate manner. One lesson from our first interview was that one of the challenges in the interviews is to find a means to communicate an apparently simple thing: what usability is, and what it is not.

Bibliography

AUCELLA, A. (1997). Ensuring Success with Usabilily Engineering. *Interactions*, May+June. pp 19-22.

BIAS, R. & MAYHEW, D. (1994). *Cost-Justifying Usability.* Academic Press.

BLOOMER S. & CROFT R. (1997). Pitching Usabilty to Your Organization. *Interactions,* Nov Dec 1997. pp 18-26

BROWNE. (1998). *In Politics of Usability*. Springer-Verlag. Berlin.

EARTHY, J. (1998). *Usability Maturity Model: Human Centredness Scale. INUSE Deliverable D5.1.4s*. http://www.lboro.ac.uk/research/husat/eusc/index_r_assurance.html

EARTHY, J. (1998). *Usability Maturity Model: Processes* (INUSE Deliverable D5.1.4p). http://www.lboro.ac.uk/research/husat/eusc/index_r_assurance.html

FELLENZ, C. (1997). Introducing Usability into Smaller Organizations. In *Interactions*, Sept+Oct, pp 29- 33.

KUUTTI, K., JOKELA T., NIEMINEN, M., JOKELA, P. (1998). Assessing human-centred design processes in product development by using the INUSE maturity model. *Proceedings of Analysis, Design and Evaluation of Man-Machine Systems*. Kyoto, Japan.

NORMAN, D.A.. (1999). Organizational Limits to HCI. Interview at CHI99, Pittsburgh.
Tudor, L. 1998. Human Factors: Does Your Management Hear You? In *Interactions*, Jan Feb 1998, pp 16 – 24.

Facilitating Communication during User-Centred Design

Masood Masoodian (masood@nis.sdu.dk)
Natural Interactive Systems Laboratory
Odense University
Odense DK-5230 Denmark

Abstract

This paper identifies some of the obstacles associated with applying user-centred design techniques to the development of generic software products for ordinary people. In particular, the problem of communication of requirements and design ideas between the users, designers, and developers is discussed. Video technology can be used to improve the effectiveness of group communication during user-centred design process. As an example a short video is described here. This video was created to combine the futuristic visions of the potential users' of a virtual meeting environment, called Magic Lounge, with their current work practices. The usefulness of this kind of video, which combines future and present use-scenarios, as a tool for generating and refining user ideas, as well as assisting communication between the members of a user-centred design team is also discussed.

Keywords

User-centred design, participatory design, scenario-based design, requirements analysis, design tools

Introduction

The aim of the Magic Lounge[*] research project (Bernsen *et al.*, 1998) is to develop and study a virtual meeting environment in which *ordinary people* can meet to work and communicate with each other using various *heterogeneous devices* such as PCs, PDAs, Palmtops, and mobile telephones. Although currently there are a number of systems which support interaction between physically remote people, the majority of these have been designed for *specific groups of users*, relying on the use of *specific computer technology*.

[*] The Magic Lounge project web site is located at: http://www.dfki.de/imedia/mlounge/

Development of systems such as the Magic Lounge, which rely heavily on the involvement of the potential users, are often based on the user-centred design techniques. However, the success of the user-centred design process depends very much on the effectiveness of the communication between the users, designers, and developers of the new technology (Kensing & Munk-Madsen, 1993). Unfortunately, in practice, there are often problems associated with this communication. Since the development of the Magic Lounge is based on user-centred design principles, it has been necessary to take a number of steps to avoid possible problems with communication between those who are involved in the process of design and development.

This paper discusses how a short video was created to combine the visions of the potential users of the Magic Lounge regarding its possible uses after the development, with their current everyday activities. Based on the evaluation of this video, a number of conclusions are drawn about its effectiveness as a tool for facilitating the communication of user requirements and design ideas between the users, designers, and developers during a software development project which is guided by user-centred design methodology.

The process

Kensing and Munk-Madsen (1993) point out that during the participatory design process, users and developers discuss issues relating to: (*i*) the users' present work practices, (*ii*) technological options, and (*iii*) the new system to be developed. Although the emphasis of Kensing and Munk-Madsen is only on participatory design process, it is reasonable to say that the effective discussion of these issues is also critical to the success of any other type of user-centred design process.

A number of techniques have already been developed to assist users, designers and developers to achieve a common understanding of the above mentioned issues. These techniques range from conducting open-ended interviews, to using questionnaires or holding design workshops. However, the effectiveness of these techniques on their own is greatly reduced, when they are applied outside an organisational setting, where the work practices of the different members of the selected user population can be very diverse.

Magic Lounge is an example of such a project. The objective of the Magic Lounge project is to develop a virtual meeting environment for *ordinary people*, who collaborate with one another on *ordinary tasks*. However, identifying the user requirements of such an environment is not trivial. Ordinary people, who may not have much experience in using technology, often don't know what a desirable system should provided. These people, on the other hand, often know what kind of tasks they would like to perform. Therefore, in the Magic Lounge project, it was decided that a scenario-based approach (Kaindl, 1995) should be used for collecting the user requirements, as this type of approach encourages the users to focus more on what they want to use the system for.

Gathering the user requirements in the Magic Lounge project started by giving the users a questionnaire to complete (Bernsen & Dybkjær, 1998; Masoodian & Cleal, 1999). In this questionnaire the users were asked to consider a use-scenario of their own (something from their daily activities), and then reply to the questions of the questionnaire in relation to their selected scenarios. Although, the questionnaire generated valuable results (Bernsen & Dybkjær, 1998), it also demonstrated that the users' expectation of the Magic Lounge varied considerably. In fact, different users had a number of conflicting requirements.

It seemed therefore necessary to conduct a series of open-ended interviews with the users, in an attempt to clarify the overall understanding of their requirements and design ideas (Masoodian & Cleal, 1999). Once again, the interviews were based on the scenarios which the users had chosen. The interviews, which were videotaped, were conducted at the users home environment. The results of the interviews indicated that even though the users had a similar "ideal system" in mind, their expectations from such a system were different; mainly due to the differences between their selected scenarios, and the differences between their knowledge of the technological possibilities. Therefore it was decided that there was a need to combine all of the user ideas within a single scenario, and then ask the users to refine and expand their vision of the Magic Lounge, using this selected scenario. It was also decided that this scenario could be communicated to the users through an audio-visual medium, such as a short video.

In recent years video has often been used as a participatory design tool, for capturing current work practices within organisations (see Gulliksen *et al.*, 1999). There are also a number of reported cases in the literature which clearly show that viewing videotaped work practices by a group of users and designers can help to create a mutual understanding of the user requirements (Brun-Cottan & Wall, 1995; Chin *et al.*, 1998). However, in all of these cases the users have come from a single organisation where they perform a particular task, or at least similar tasks. This is not the case in the Magic Lounge project, as there is no single organisational setting or work practice.

Due to these reasons, a short video (10 minutes) was made to combine all of the user requirements and ideas, which were identified through the user questionnaires and interviews. A single realistic meeting scenario was selected as the basis for the story of the video. In this scenario, a number of physically remote people join together in the Magic Lounge for a Marine History Club meeting. To make the video even more realistic for the users, a major portion of it was cut from the video recordings that were made during the interviews with the users. The end result was a film that combined facts (what users do currently) with fiction (what users would like to do in the future).

The video was then shown to some of the potential users of the Magic Lounge during a workshop. The users' response to the video was very positive. They claimed that seeing the video helped them visualise what the Magic Lounge could be used for after its development. It also assisted the users with providing the design team with new ideas, as well as clarifying the designers' understanding of the users' old ideas.

Discussion

The problem of facilitating communication between users, designers and developers, as described in the previous section, is common to many user-centred design and development projects (see Gulliksen *et al.*, 1999). However, this problem becomes even more challenging to overcome when the user-centred design methodology is applied within a research project which aims at developing an innovative system for which there are no currently available use cases. In fact, often in these types of projects if not enough attention is given to fostering effective communication between those involved in the process, user-centred design can gradually turn into technology-driven design, where the role of the users is almost completely ignored.

One of the reasons for possible communication breakdowns during the user-centred design process, when it is applied to developing innovative technologies, is due to the inherent *iterative* nature of the user-centred design. In such projects, during the early stages of software design the users are generally inquired about their work practices, requirements, expectations, and design ideas. The findings of these inquiries, which can be in the form of interviews or questionnaires, are then combined to create the initial design solutions, which are subsequently refined or changed after user evaluation. However, if an iteration of this process takes a long period of time, as it does in many projects, then the users are less likely to get enough feedback from the designers regarding their design contributions to make them feel as being valuable and equal partners in the project. This is particularly true when the users are not aware of the fact that software design often requires making various kinds of tradeoffs, which means that not all of the user ideas may end up being part of the final design.

An obvious solution to this problem is to speed up the process of iteration, either by taking smaller iterative design steps, or by providing different kinds of feedback to the users during the larger iterative steps. Creating simple and quick prototypes is one way of achieving this objective. It is also possible to use video technology for providing effective feedback. A short video, such as the one described in the previous section, can be created very quickly to help the users see the effects of adopting various design options. This can also help the individual users to compare their own ideas against those proposed by the others, so that if their own ideas are not adopted then they are less likely to feel that they have been ignored.

Conclusions

This paper described the importance of facilitating effective communication between different members of a user-centred design team. There are a number of ways in which this communication can be improved, one of which is through the use of video. Video is a powerful tool for creating mutual understanding of the consequence of adopting various design options, as well as communicating the tradeoffs associated with different design ideas. In many ways, a video can be considered as a complementary tool to ordinary software prototyping.

However it is important to note that when this type of video is created, it should be based on the users' existing ideas rather than the designers' ideas, otherwise the video itself might influence the users' future ideas to reflect more what the designers had in mind.

During the workshop it would be very valuable to spend some time discussing the participants' experiences in using tools, such as video, software prototypes and mock-ups, for facilitating communication process during the user-centred design-based system development projects.

Acknowledgements

Magic Lounge is an Esprit Long-Term Research project (No. 25458). The author would like to acknowledge the contributions of the other Magic Lounge project members, in particular Bryan Cleal, who with the participation of the Magic Lounge user group, created the short video.

References

BERNSEN, N. O., RIST, T., MARTIN, J-C., HAUCK, C., BOULLIER, D., BRIFFAULT, X., DYBKJÆR, L., HENRY, C., MASOODIAN, M., NEEL, F., PROFITLICH, H-J., ANDRE, E., SCHWEITZER, J., & VAPILLON, J. (1998), Magic Lounge: A Thematic Inhabited Information Space with "Intelligent" Communication Services. *Proceedings of NIMES '98*, 188-192.

BERNSEN, N. O., & DYBKJÆR, L. (1998), Dimensions of Virtual Co-presence. *Proceedings of COOP '98*, 103-106.

BRUN-COTTAN, F., & WALL, P. (1995), Using Video to Re-Present the User. *Commun. ACM 38 (5)*, 61-71.

CHIN, G. JR., ROSSON, M. B., & CARROLL, J. M. (1998), Participatory Analysis: Shared Development of Requirements from Scenarios. *Proceedings of CHI '98*, 162-169.

GULLIKSEN, J., LANTZ, A., & BOIVIE, I. (1999), User Centered Design – Problems and Possibilities, A Summary of the 1998 PDC & CSCW Workshop. *SIGCHI Bulletin 31 (2)*, in print.

KAINDL, H. (1995), An Integration of Scenarios with their Purposes in Task Modeling. *Proceedings of DIS '95*, 227-235.

KENSING, F., & MUNK-MADSEN, A. (1993), PD: Structure in the Toolbox. *Commun. ACM 36 (4)*, 78-85.

MASOODIAN, M., & CLEAL, B. (1999), User-Centred Design of a Virtual Meeting Environment for Ordinary People. *Proceedings of HCI International '99*, in print.

Author

Masood Masoodian is an Assistant Professor at the Natural Interactive Systems Laboratory, Odense University, Denmark. He has a Ph.D. in the area of Computer Supported Co-operative Work. Over the past few years, Masood has been doing research on design, development, and evaluation of collaborative work environments. He has also carried out a number of empirical studies, focusing on the role of different human-to-human communication media in supporting group interaction in synchronous computer-based shared workspaces.

A Case Study of User-Led Systems Design and Development in Healthcare Informatics

Rob Procter[1] (rnp@dcs.ed.ac.uk), Mark Hartswood[1], Michael Sharpe[2] & Alan Doris[2]
[1]Institute for Communicating and Collaborating Systems, Division of Informatics, University of Edinburgh
[2]Departments of Psychiatry and Psychological Medicine, University of Edinburgh

Introduction

Numerous prescriptions for making IT systems design and development methodologies more “user-centred” have been put forward in recent years. While such initiatives are welcome, in as much as they continue to privilege the role of the designer, they leave important issues unaddressed. In particular, even user-centred methodologies often remain essentially Òtop-downÓ in character with only limited opportunities for user involvement. As a consequence, requirements and usability problems that can only be identified in the context of use are missed. These are serious deficiencies, but the most critical failure of most methodologies is their inability to exploit users’ own capacity for innovation (Adler and Williams, 1991).

Recent studies of IT design and development practices suggest that users are increasingly able to take on these roles for themselves (Procter, Williams and Cashin, 1996b). In part, this is because many technologies are now available in the form of generic components that non-specialists can customise and configure on a “pick and mix” basis (Procter and Williams, 1996a). With the right technologies, design and development can take on the character of “bricolage“ -- the assembly and configuration of “bits and pieces” of software and hardware (Buscher, Mogenson and Shapiro, 1996).

In contrast to user-centred approaches, such “user-led” practices enable users to achieve a direct role in design and development in their own work settings and exploit opportunities for social learning -- the sharing of ideas, experiences and innovations -- between individuals and small groups (Procter, Williams and Cashin, 1996b). Users’ requirements are more likely to be met from the start and to continue to be met even as the interplay between new technology and work practices leads to re-formulation (Coble, Karat and Kahn, 1997): “design in use” (Greenbaum and Kyng, 1991) becomes a practical proposition. We argue therefore that, rather than being merely user-centred, IT systems design and development should also seek to be user-led.

Our work sets out to explore the viability of user-led design and development practice in the context of a large organisation. In pursuit of this goal, we are executing a case study of a user-led design and development project in a healthcare setting. Wi thin many organisational settings the risks of uncoordinated user-led development which may be perceived by IT services management to outweigh its benefits (Jakobs, Procter and Williams, 199 6). For example, it may lead to interoperability problems between heterogeneous systems. As a consequence, it is important that user-led developments be maintained in alignment with the broader, strategic concerns of IT services management.

In studies of the financial sector, Procter et al. (1996b) observed the emergence of new, specialist groups within IT departments working closely with users and acting simultaneously as facilitators and gatekeepers of technical change. One of our main goals is to explore whether such models for the management of user-led design and development are transferable to different organisational contexts.

The project

The project focuses on the design, development and use an electronic medical record (EMR) system for healthcare. In principle, EMR offers clinicians the potential for instant, location independent access to comprehensive and integrated patient data. In practice, however, the realisation of these benefits has proven very difficult (Kushniruk et al., 1996). Although there has been considerable progress in the development of EMR system infrastructures, there has been much less progress in tackling usability issues. In many respects, these problems are symptomatic of those experienced within the wider healthcare informatics setting, and can be attributed to the lack of clinician input into system design and development (Heathfield and Wyatt, 1993). This leaves designers and developers without an adequate understanding of clinicians' needs and of important usability problems.

Deliberate Self-Harm (DSH) was chosen as the project setting chosen because it has several features that make it particularly likely to benefit from improved information handling:

1. a high number of emergency admissions,
2. a short average time in hospital making fast access to records essential,
3. a high rate of re-admission to the same service, quick access to previously recorded records being necessary, and
4. the patients are high users of other health and social services, requiring rapid communication of information gathered at assessment to other agencies.

Potentially useful technologies for EMR in the clinical setting -- e.g., speech recognition, mobile computing -- are now becoming widely available (e.g., Lai and Vergo, 1997; Miah and Bashir, 1997). The goal of the project is to explore how usable configurations of such technologies -- and practice innovations -- can emerge through user-led processes of design, development and evaluation (Procter et al., 1996b). Most clinicians wor k in a demanding environment which leaves little time to develop and practise IT skills. To address this, the project makes available to clinicians the services of a "facilitator" who provides them with requisite technical skills on site. Through immers ion in the work setting as a participant-observer, the facilitator acquires an understanding of the work that enables him to help clinicians to translate their ideas into a useful and usable system.

Methodology

Our methodological approach is informed by the earlier studies of Procter et al. (1996a; 1996b) and by the work of Buscher et al. (1996). A key goal is to situate design and development in the workplace. In this way, the project seeks to address the limitations of more conventional user-centred methodologies. Members of the project group work closely with clinical staff at all times. One of the project team acts as technical "facilitator" to the clinicians. His role is to carry out the investigation of work practices, to work with clinicians to explore and draw up requirements for EMR, to implement prototypes and evaluate them.

Ethnographic methods for the study of work are finding an increasing role within medical informatics (Friedman and Wyatt, 1996) and IT systems development generally (Hughes, Randall and Shapiro, 1992). We use these methods -- including interviews, observation and document analysis -- to gain a rich understanding both of clinicians' current work practices and of how they change with the introduction of EMR. In this way, w e aim to uncover tacit assumptions that are not available to casual inspection, tease out usability requirements, establish metrics for evaluating EMR and assess its impact on practice. Interviews and discussions with clinicians are recorded and notes ma de of activities observed and artefacts employed -- e.g., clinicians' notes, records, referral letters.

For the purposes of system design and development, the emphasis is on "lightweight" design, prototyping and evaluation techniques where rapid results are as important as technical sophistication. Techniques employed range from informal discussions to more formal co-operative design and evaluation "workshops". Activities are tightly coupled: Buscher et al. (1996) describe a methodology in which "the effort shifts fairly smoothly between implementing or adjusting previously decided possibilities, picking up on the host of small problems that arise during work, coping with the unanticipated consequences of previous actions, talking to individuals, and occasion ally setting up larger meetings for important decisions". Extensive use is being made of off-the-shelf, configurable technologies and devices such as palm tops, personal digital assistants (PDAs) etc., which are customised to create prototypes for use by clinicians. Prototyping draws upon a number of techniques, ranging from storyboards and paper-based mock-ups to partial implementations. These are evaluated by clinicians and rejected or refined through further iterations.

Summary

The aim of our work is to contribute to understanding the form and relevance of workplace based, user-led design and development methodologies within an organisational context. The issues we are addressing include: the form, content and timings of the contributions users are able to make -- both as individuals and collectively -- to design and development ; the barriers -- if any -- to their effective involvement; the problems of reconciling different working practices; developing an understanding of how to align user-led design with the wider, strategic concerns of information services management; and th e problems of maintaining this alignment over time.

Acknowledgements
We would like to thank the staff members of the DSH clinic, Royal Infirmary of Edinburgh for their help and co-operation in this project. This work is funded by the Engineering and Physical Sciences Research Council, grant number GR/M52786.

Bibliography

ADLER, M. & WILLIAMS, R. (1991). The Social Implications of the DSS Operational Strategy. *Social Policy Series No. 4*, The University of Edinburgh.

BUSCHER, M., MOGENSEN, P. & SHAPIRO, D. (1996). Bricolage as a Software Culture. *4th Software Cultures Workshop,* Technical University of Vienna, 1996.

COBLE, J., KARAT, J. & KAHN, M. (1997). Maintaining a Focus on User Requirements Throughout the Development of Clinical Workstation Software. In *Proceedings of CHI'97*, ACM Press.

FRIEDMAN, C. & WYATT, J. (1996). *Evaluation Methods in Medical Informatics.* Springer.

GREENBAUM, J. & KYNG, M. (1991). *Design at Work: Co-operative Design of Computer Systems*, Lawrence Erlbaum.

HEATHFIELD, H. & WYATT, J. (1993). Philosophies for the design and development of clinical decision-support systems. *Methods Inf. Med., 32(1):* 1-8.

HUGHES, J., RANDALL, R. & SHAPIRO, D. (1992). Faltering from Ethnography to Design. In *Proceedings of CSCW'92.*

JAKOBS, K., PROCTER, R. & WILLIAMS, R. (1996). Non-Technical Issues in the Implementation of Corporate Email: Lessons From Case Studies. In *Proceedings of the ACM SIGCPR/MIS Conference.* ACM Press.

KUSHNIRUK, A. ET AL (1996). Assessment of a Computerised Patient Record System. *MD Computing, 13(5).*

LAI, J. & VERGO, J. (1997). MedSpeak: Report Creation with Continuous Speech Recognition. In *Proceedings of CHI'97,* ACM Press.

MIAH, T. & BASHIR, O. (1997). Mobile workers: access to information on the move. *Computing and Control Journal*, October, pp. 215-223.

MCDONALD, C.J. (1997). The Barriers to Electronic Medical Record Systems and How to Overcome Them, *JAMIA , 4*:213-221.

O'NEIL, M., PAYNE, C. & READ, J. (1995). Read Codes Version 3: A User Led Terminology, *Meth. Inform. Med., Vol. 34*: 187-92.

PROCTER, R. & WILLIAMS, R. (1996a). Beyond Design: Social Learning and CSCW -- Some Lessons from Innovation Studies. In *Shapiro, D. et al (Eds.), The Design of CSCW and Groupware Systems*, Elsevier Science, pp. 445--463.

PROCTER, R., WILLIAMS, R. & CASHIN, L. (1996b). Social Learning and Innovations in Multimedia-based CSCW, *ACM SIGOIS Bulletin, December*. ACM Press, pp. 73-76.

The authors

Rob Procter is a senior lecturer in the Division of Informatics at the University of Edinburgh, one of the leading UK centres for informatics research. Dr. Procter's research speciality is Human Factors. He holds, or has held grants, from the SERC/ESRC Joint Committee, the EPSRC, the Scottish Higher Education Funding Council and the EC Telematics Applications Programme.

Mark Hartswood is a Human Factors researcher in the Division of Informatics at the University of Edinburgh, with extensive experience of healthcare informatics. His research has included field studies of work practices at UK breast screening clinics, studies of the eff ects of computer-based aids on radiologists' decision-making and clinical evaluations of IT systems.

Michael Sharpe is a senior lecturer in psychological medicine in the Department of Psychiatry, University of Edinburgh, which has a long history of research in the epidemiology of deliberate self-harm (DSH), and an honorary consultant at the Royal Infirmary of Edinburg h.

Alan Doris is a Medical Research Council clinical scientist and honorary specialist registrar in the DSH service within the Department of Psychological Medicine at the Royal Infirmary of Edinburgh.

Creating a User-focused culture in an Internet company

Rajhev Rajkumar (rajhev@iii.co.uk

Background

Interactive Investor is a company that exists solely on the Internet, offering financial and investment information to investors in South Africa, The UK and Asia. The site helps you learn about finance and investments, investigate and analyse financial products, trade, and monitor your investments. It has enjoyed unprecedented growth, especially in the UK, and is set to enjoy even greater success with its aspiration to look beyond the sophisticated investor and target the unaware investor as well. Hitherto it has held a handful of focus groups, beta-tested two product releases with regular users, and more recently, started to conduct empirical testing in trying to understand the users of the Web-site.

Challenges

Time and resources: For a company existing on the Internet, there is a considerable advantage to being first to market with new or enhanced products and services. This agility often means tight delivery deadlines resulting. The trade-off is that it becomes difficult to involve users in the design process due to lack of time and resources.

Buy-in from designers: The success of the site thus far despite it's lack of a customer-centred design process means that it becomes increasingly difficult to convince the development team of the importance and need for introducing the user into the design and development process. There is also the issue of designers who believe that their knowledge of the investment arena qualifies them to represent our entire target market, and, on the other hand, designers who believe that they can architect information despite a lack of understanding of the financial domain.

Wide range of users: There is also the challenge of finding a representative sample of users when the target market covers a range of users varying from ignorant to expert in their knowledge of the medium (Internet) and domain (financial services and investments). This is further complicated when one considers the wide variety of financial needs that the site has to try and satisfy, and the fact that the different sites have to try and maintain consistency while targeting users from three diverse regions.

The way ahead

Since this is a relatively new initiative, the following are a few of the plans in the pipeline to make users an integral part of the design process.

Recruiting users: Use the site to recruit a pool of users for qualitative research and empirical testing in London and other major cities in each of our markets.

Online feedback: Recruit users who want to play an active role in the design process by informing us of their needs and preferences via email and online polls and surveys. This group will also help to beta test products, and their behaviour will be analysed using the log files of their movements around test sites.

Knowledge sharing: Work is also beginning on a usability Intranet to share and discuss findings and results, and to act a pool of knowledge and resources on usability. Developers will have access to video highlights of usability tests and focus groups as soon as a usability lab has been commissioned, as well as the opportunity to meet customers and financial intermediaries for one-to-one interviews and briefings. A usability mailing list is being set up for those who wish to share ideas, debate about findings, etc.

Commitment to customer centred design: Service level agreements are being drawn up between the various stakeholders (management, designers) and the usability team to obtain commitment that the design process will take the user into consideration when and where necessary. Quality standards and interface design guidelines are also being put in place.

Exploring new techniques: The design team is being educated about alternative approaches to design, such as iterative design, parallel design, participatory design and paper prototyping. These approaches have been used to a very limited extent in the past, if at all.

Biography

Rajhev Rajkumar, 1969, did his BSc in Mechanical Engineering in 1992 at the University of Natal, Durban. After working for two years in an engineering environment, first in a sales-engineering role and later in a maintenance-engineering role, he undertook a change of career into advertising copywriting and scriptwriting. It was during this period that he was first exposed to the Internet, gaining experience in writing copy specifically for the Web and also in creative conceptualisation for Web-advertising. He subsequently started to work full-time on the Internet, first as a site editor, then as a strategic consultant and manager of a Web-development team. He currently works as a Business Consultant for Interactive Investor International, a company focused on helping investors in the UK, Asia and South Africa to manage the finances over the Internet. In his 2 years of experience with working on the World Wide Web, Rajhev Rajkumar has had experience and exposure to Web-marketing, strategic planning, content architecture and editorial management, and most recently has started an initiative to create and champion a user-centred design process at Interactive Investor.

Using UCD for the Web – a case story

Hans Erik Sørensen (hes@daimi.au.dk)
Department of Computer Science, University of Aarhus
Ny Munkegade 540, 8000 Århus C., Denmark

Introduction

A corner stone in user centred design is to carefully identify and define the specific needs and characteristics of the user population as well as the characteristic of the context(s) of use. This central task has become difficult to carry out when designing internet sites. People from all over the world have access to a site and may have numerous reasons for surfing to it.

This large heterogeneous user population is in itself problematic, but when you are facing a very limited time frame, in addition, it seems almost impossible[xiv]. The only way to address this problem, and still use UCD, is to use quick methods that involve a lot of people. It is questionable if these methods collect enough information to base design on.

The case

In order to do users centred design in a realistic context I joined a project of a major Danish public sector software supplier. The aim of the project was to design a skeleton for the Danish municipalities sites containing services from all levels of administration. The time frame of the project was approximately 3 months from we entered until the final implementation.

Fieldwork

We started out by conducting a two day field study in three municipalities with different demographic conditions. Doing the field-study we interviewed the employees at the city hall and other places, where citizens have access to public services, about the nature of the questions and requests. We also talked to citizens about their needs and experience in relation to public services.

While conducting the fieldwork we collected artefacts representing public services which could be used later on.

[xiv] In most cases I would argue that it also is impossible.

Card-Sorting Workshop

Following the fieldwork we invited 24 citizens to join 3 workshops (8 citizens per workshop), where we wanted the citizens to tell us how they thought the services should be organised.

In advance they were asked to remember their resent personal interactions with the public authorities. These were then explained to the rest of the participants of the workshop and represented on a piece of paper. On basis of the field study we had created some scenarios that were also presented and written down.

All the papers were then organised through co-operative card-sorting. After finishing organising the pieces of papers into a labelled structure, the artefacts collected while carrying out the field study were presented. The items that the citizens thought would be interesting to have access to through the internet were incorporated into the structure.

Remote formative evaluation

Hands-on experience in familiar situations is considered to be necessary to support reflection and design by users [Ehn & Kyng 1991]. We therefore fed the knowledge we had gained from the previous events into a prototype, and had a large set of users (24) carrying out tasks that were sent to them. The tasks were represented as small textual scenarios that could set the stage for action [Bødker 1999] and were created on basis of narratives that were collected doing the previous events. The fact that we created the scenarios implies that we could not ensure, that the users had been in similar situations before, but we hoped that the scenarios would still constitute familiar situations.

While the users were carrying out the task, their communication with the web-server was logged, i.e. their actions on the site were stored. After completion of each session we had them write a diary about their use and impressions. This combination of direct and indirect data of the users interaction with the site was analysed through a graphical representation of the data - "Use Case Signatures"- that was designed during the project [Kaasgaard et. al. 1999 [xv]]. The data contained not only impressions and use of the site but also design suggestions. The new knowledge was then fed into the prototype.

This evaluation process was iterated 3 times with an interaction cycle of a week. We did not do further iterations because of time constraints.

Discussion Workshop

After the evaluations we still felt that we did not know enough about why some users used the site as they did. We therefore arranged a new workshop that was not scheduled from the start. To this workshop we invited some of the most actively engaged users during the process and the designers, so that they could discuss their problems without having everything mediated through us. All parties had in advance prepared things that they wanted to discuss with the rest.

[xv] The paper do also describe other parts of this project in more detail.

The discussion went on for 3 hours without the need for control. Most of the discussion was about how the prototype was used and how it should evolve.

Prototypes as boundary objects

Star & Griesemer have the notion of a boundary object as something "which would maximize both the autonomy and communication between worlds." [Start & Griesemer 1989 p. 404]. This is also what we need in UCD; something that each group participating in the project can utilise, and that can be shared between groups while still maintaining a common identity. Doing the project we heavily used prototypes as boundary objects between all groups involved in the project.

The use of prototypes in the project

The groups participating in the discussion workshop had already used the prototype in advance for different purposes:

- Designers and engineers used prototypes in order to concretise and embodying ideas, hereby having the chance to detect problems with the ideas.
- Users used the prototypes to get hands-on experience hereby allowing them to evaluate the concepts and ideas.
- UCD facilitators (HCI people) used the prototype to design tasks and for understanding the users responses.

Unlike requirement specifications and other objects that are only usable for one group, the prototypes were familiar to all. In communication there is a need for a some common knowledge in order for people understand each other. In this situation the familiarity with the prototype gave the participating groups this common knowledge, i.e.. it gave them a reference object. The communication then led to increased knowledge of the contexts. The designers and HCI people learned about the use and use situations, and the users learned about the limitations and possibilities of the internet, which made them capable of contributing with realistic design suggestions.

We used the prototype not only between users, UCD facilitators and designers but also towards management. From the prototypes they got an understanding of how the design evolved, and where it was heading. The knowledge was used to decide if there were a need for collaborating with other service providers, and to make other decisions about the future of the project.

Limitations

Although the use of prototypes as a boundary object in this case was fairly successful there may be limitations. Resent research in scenarios [Bødker 1999] has shown that in order to make scenarios really usable, they must be constructed with the specific use in mind. It seems that in larger projects than web-design this is also the case for prototypes.

Scenarios and prototypes have some common features, due to the fact that they are both used as parts of creating a common information space. When using objects of a common information spaces outside groups with shared understanding of context "... there is a much greater need for refining and "packaging" information into a meaningful context, in order to maximise the likelihood that the intent of a message is received appropriately, and the recipient is also required to expend some effort in order to "unpack" this information..." [Bannon & Bødker 1997].

The implication of this for prototypes may be, that either the developers of the prototype and/or the receiving users have to do some extra work in order to create the basis for a shared undestanding. Focusing a prototype towards the needs of one communication is applicable, but "packing" enough context into a prototype for it to support communication between all groups may in larger projects not be practically possible. On the other side, if the prototype is not "packed", users from other groups are forced to use it in a context where it is not intended. Most of the problems encountered with the use of prototypes may be due to the fact that prototypes created to suit the needs of one communication is used in connection with another. An example is that designers at a company happily showed their prototypes to peers but avoided showing them to management and executives because "Good ideas may be rejected by ill-informed executives based on what is perceived as inadequate execution of the prototype." [Schrage 1996, page 200]. The prototype was created to support the communication in the design group, and was probably successful, but when used in another setting without "packing" it to suit the needs of that group - it failed.

The result of this may be that it is not possible to create a boundary objects that can be used by all groups, but by constructing specific prototypes the increased communication inside one or between specific groups may compensate for the drawbacks of using prototypes as the basis for a shared understanding of context.

Conclusion

The case showed that it is possible to use quick methods in connection with large heterogeneous user population, and still attain enough information to design a relatively simple interface as a web-site. I am under no circumstances claiming that the site is usable for all people, but I think that we based the design on information from a lot of people considered the time and resources at hand.

On the success of the heavy use of prototypes in relation to all involved groups, one important question, is if it is a result of the nature of this project, or is the prototype in general the boundary object that is able to suit the needs of all the groups, and at the same time makes us capable of communication across all borders? If it is not; is the only possibility for creating a shared understanding of contexts then to work together co-operatively?

Acknowledgements

The case described in this paper was carried out as part of the BIDI project (Usability Work In Danish Industry). BIDI is sponsored by the Danish Center for IT Research through grant no. 23.

References

BANNON, LIAM & BØDKER, SUSANNE (1997): “Constructing Common Information Spaces”, in *Hughes, J., Prinz, W., Rodden, T. & Schmidt, K. (Eds.): Proceedings of ECSCW97*, Dordrecht: Kluwer, pp. 81-96.

BØDKER, SUSANNE (1999): "Scenarios in User-Centred Design - setting the stage for reflection and action", in *Proceedings of HICSS'99.*

EHN, PELLE & KYNG, MORTEN (1991): “Cardboard Computers: Mocking-it-up or Hands-on the Future”, in *Greenbaum , J. & Kyng, M. (Eds.): “Design at work”*, LEA, pp. 169-195.

KAASGAARD, KLAUS; MYHLENDORPH, THOMAS; SNITKER, THOMAS & SØRENSEN, HANS ERIK (1999): "Remote Usability Testing of a Web Site Information Architecture: "Testing for a Dollar a Day"", to appear in *Proceedings of Interact'99.*

SCHRAGE, MICHAEL (1996): “Cultures of Prototyping”, in *Winograd, T. (Eds.): “Bringing design to Software”*, AddisonWesley, pp. 191-205.

STAR, SUSAN L. & GRIESEMER, JAMES R. (1989): “Institutional Ecology, ‘Translations’ and Boundary Objects: Amateurs and Professionals in Berkeley’s Museum of Vertebrate Zoology, 1907-39”, in *Social Studies of Science*, vol. 19, SAGE, pp. 387-420

On discerning users

Harold Thimbleby (harold@mdx.ac.uk)
Middlesex University, Bounds Green Road, London, GB

Introduction

User centred design works on behalf of users engaging processes that improve quality of system design, particularly with the aim of improving usability. This position paper argues that until users as a group have empowered discrimination in usability issues there will be negligible impact on manufacturing and production processes.

We make three main points:

- Problems with usability are cultural. Usability is not glamourous for manufacturers, and users are often "fashion victims" who do not and are not able to make choices informed by usability, even when it is in their own interests to do so.
- Design tools should be built that are available on the World Wide Web that would permit widespread collaboration. This would give users (including interace critics) informed access to early stages of design, as well as the usual leverage of user groups on the Internet. Wider dissemination of clear usability issues will expose both good and bad practice.
- "Public understanding of science" should be broadened to encompass the science necessary to understand the basic workings of interactive devices — this should cover both cognitive engineering issues as well as computing science issues. We see the users understanding what usability is about as essential to successful user centred design.

This position paper describes the culture accepting unusability as our own responsibility. It then makes a positive suggestion: to promote public understanding of science, particularly through good demonstration systems, with good user interfaces.

unusability CULTURE

Although unusabilty is bad in almost every situation, users seem to accept it, and manufacturers continue to make unusable systems. Why is there this unusability culture, and why is it so persistent?

Video recorders, aircraft flight decks, and desk top computers are all examples of complex interactive systems. Interactive systems are central to modern life, but there is widespread dissatisfaction with even 'simple' systems: they are too often awkward and frustrating to use. Poor design, including poor user manuals, have been blamed for accidents — about 90% of all computer-related deaths are caused by poor user interface design (MacKenzie, 1996). Despite the 'obvious' bad design, there is widespread acceptance for the problems, both from users and professionals.

Human society is complex, and although there is no *a priori* reason why society should be complex, we are embedded within it and in principle cannot understand it reliably. Society in fact is at the limits of human abilities to comprehend. We could argue, for instance, that our educational system has a hard job succeeding, and there are many failures — not least because our technology is so complex. Indeed, modern technology is made by people who have succeeded through a long period of education, and stand, as it were, at the pinnacle of the educational pyramid. The few sufficiently-skilled people design and build computer systems, and almost inevitably, because they do the best they can, what they do is hard to comprehend for the many other people less skilled than they.

Exacerbating the complexity: the people who use systems are themselves in society, and therefore push their own limits. Businesses are competitive and try to out-do each other — hence relying on exploiting skills they hope other businesses cannot use so well. Thus computers-in-use present an even more complex situation than the already complex-enough society.

Complex systems are hard to describe. That user manuals are often wrong in detail suggests that manufacturers even find their own products hard to describe! If we can't say exactly what a video recorder or a word processor does, how are we going to improve it?

The following sections (§3, 4) by no means cover the full range of issues. Other relevant discussions are Norman (1998) and Thimbleby (1990a, 1990b, 1992, 1993, 1996, 1998a). A range of non-HCI issues are covered in: Christensen (1997) which provides a business perspective — don't listen to customers; Paulos (1990) is a mathematical perspective — users and designers are innumerate; Piatelli-Palmarini (1994) is a psychological perspective — none of us make sensible decisions, either to design or to consume; and Tenner (1996) provides a pessimistic "any advance has problems" perspective!

Culture affects users

HCI commodified

Many user interfaces, about which HCI has a proper concern, are also consumer items. We buy word processors, email systems, video recorders, and so on. Thus in many user interfaces, we have an actual stake in the outcome of interaction. Few people would be happy admitting they made a financial mistake buying product *X* over product *Y*. Moreover, reducing cognitive dissonance means that people will be tempted to rationalise any difficulties they do have! Thus any branch of HCI that takes as its starting point the possibility, the desirability, of changing the *status quo* has a deeply embedded cultural hurdle to overcome.

In everyday life, when HCI problems are experienced, we have been trained to take responsibility for those problems. We can buy help books, or computer upgrades, or the market will offer new technologies providing new features and 'solutions.' In every case, fixing HCI problems has practically become a consumer pastime, though disguised as buying into better systems — as fashion, in fact. Behind the fashion is the cultural view that usability problems are the user's problems, not the designer's. As user problems, the user can solve them by buying a ‘better’ system, or better documentation, or ‘upgrading’ their current one.

In all cases the responsibility for action is directed *away from* improving HCI.

Delaying quality

Norman (1998) credits Christensen (1997) for another view of “avoiding HCI” that has become endemic. Technology is getting better, and has increasing performance over time. We can represent this by the graph, below, using a line of positive slope (it is not necessary for our purposes to worry about the precise shape of the line, or exactly what ‘performance’ is measuring). For any particular task the user has, some level of performance *p* will be required. From the graph it is clear that there is a threshold point when the lines intersect, at performance=*p* and time=*t*. Before *t*, technology is delivering inadequate performance; after *t*, technology can deliver more than adequate performance. See Figure 1.

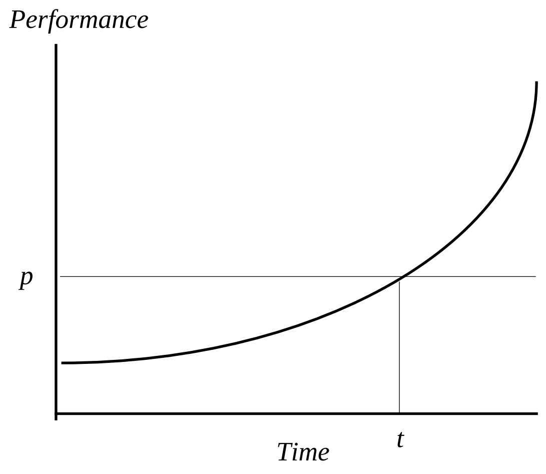


Figure 1. Perfomance of technology increases with time. At time t perfomance exceeds a threshold value p.

So before the threshold time, all manufactures have to do is promise increased technical performance (which is easy, since technology is getting better all the time). After time *t*, products get distinguished not by their performance but by more subtle — and harder to supply — properties like usability. It is therefore in manufacturers’ interest to increase *p*, because this will postpone the threshold time. For technologies like wrist watches, we are long past the threshold, and they are now fashion items. For many interactive systems, like word processors, we should also be well beyond the threshold. So why aren’t word processors much better?

The tragedy of feature stepping

The tragedy of the commons is that farmers acting in their own best interests over-graze common land. If the land is being over grazed, the community should hold back. But it is to the advantage of any individual to graze a few more of their own animals, especially if other people are removing theirs from the common land. The same effect occurs with technologies that are shared: each person benefits by having the “best,” but because not everyone can simultaneously have the best, a feature-stepping race occurs. The result is that few people ever benefit from their investment. Systems and training becomes obsolete. (The millions of tons of computers the UK landfills annually is a testament to the continual over-taking of once-leading computers.)

What we call "feature-stepping" is illustrated in the diagram below. For simplicity, assume there are just two users. One user has some required level of performance. They get a message from the other user, who is using a more advanced system, and so they are forced to upgrade. However, by the time they upgrade, the manufacturers have improved the performance of the technology, so they upgrade *beyond* the level of the other user. Then the situation between the users is reversed, and the first user wants to upgrade … and so on. Each user upgrades alternately, and if the manufacturers play the game properly, the rate of performance requirements increasing due to stepping stays above the performance the technology can deliver. Thus manufacturers can keep users permanently behind the threshold time. Based on Figure 1, Figure 2 illustrates feature stepping.

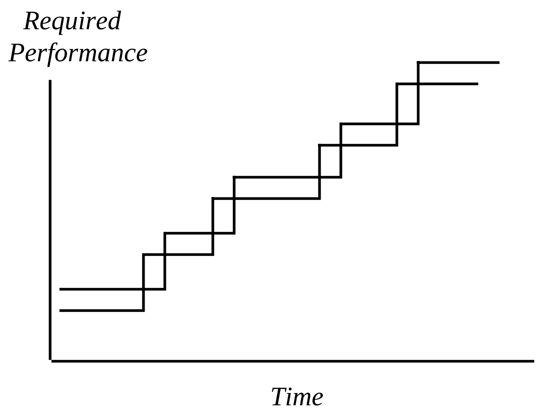


Figure 2. Schematic of feature stepping.

Manufacturers are aware of these community pressures and they may take further steps to encourage the "upgrade habit." Users of old version systems are obviously encouraged to upgrade so they can read later versions; but they may be forced to upgrade by their "richer" colleagues because later versions of the software may deliberately not be able to read older versions. In other words, both old and new users want old users to upgrade. This is a powerful way of increasing *p*, especially when many users are spread around the world, and there is no way (e.g., organisational purchase controls) of controlling their understandable urges to keep up.

Would standardisation help, so all users and manufacturers had an agreed level of performance for tasks? Possibly not. Technologies like the web are a good example of this behaviour, but with the twist that standards are set that anticipate future technologies. Naturally the World Wide Web Consortium sets standards that are the best possible — but that means they are above what most users are capable of. Thus the stepping is always above the performance available that technology delivers, and users are permanently kept hoping for the future threshold.

This seems somewhat cynical, but Microsoft (as an example of a leading manufacturer) have admitted doing it (Gibbs, 1997): they are quoted as saying, "if we hadn't brought your processor to its knees, why else would you get a new one?" This looks like stepping, pitting harware and software against each other. Once a user buys the new processor, they are in a position to put pressure on other users to upgrade to keep up with them. It is interesting that most processor purchases include bundled software: thus giving users what seems like a free upgrade. Actually, it’s marginal for them, and expensive for everyone else who they cause to step!

Later versions of Microsoft Word cannot read files saved by certain earlier versions. Thus users are faced with the choice of upgrading or being isolated — or the other user might try downgrading (and discover that de-installing has been made surprisingly tricky).

People who use the latest standards force “backward” users to upgrade. They then upgrade — but they buy into technology that is the latest, and therefore ahead of everyone else. So the co-stepping cycle goes. (Like the tragedy of the commons, each individual’s sensible behaviour is to the whole community’s detriment.)

In short, consumers of complex computer systems have been kept — for marketing reasons and so on — to the left of the threshold. Kept “in their place” their job is to consume, rather than to demand better systems, with better HCI. Nobody is critical of bad HCI, because their systems are anyway inadequate and need upgrading …

Homeostasis in effort

As technical performance improves, society’s standards for tasks will also be pushed upwards, if we assume there is a constant social value associated with the costs of performing tasks. Thus as things become easier to do, the social value of what users achieve decreases. To maintain the same social value, then users must do more sophisticated things.

Word processors illustrate this phenomenon well. Once it would have been sufficient to print text that looked written by a typewriter, but as this became easier to do, more fonts were required, then colour, then clip-art, and so on. Thus p is increased, and becomes an upward curve above the technological performance. The consequence, again, is that the time to the threshold of good HCI is postponed.

Consumer (in)action

In the 1960s, some cars were badly designed and unsafe to drive. As Ralph Nader exposed (Thimbleby, 1993), the prevailing cultural assumption was that drivers had accidents, and therefore drivers were responsible for the behaviour of cars. For example, if a parked car rolled down a hill, the driver should have applied the parking brake properly, they should have turned the wheels so the car would roll onto the kerb, and so on. That the car might have been badly engineered and have a feeble parking brake was thereby disguised.

The same sort of problem arises with HCI. Users are persuaded that their problems are their own problems, and that to use complex software they should take responsibility for their own training. Users have usability problems, *so* users should learn how to use computers properly.

The computer-based society is certainly a complex place, and people do need to be computer-literate because that is how the world is. But this practical response must not be used as an excuse to make systems more complex than they need be, because users will take it upon themselves to learn how to use them. Obviously something has to be known about computers and some training is appropriate; but at present the balance is clearly in manufacturers’ favour. Indeed, manufacturers often commodify their learning material, thus making further profit by providing systems that require additional training!

Companies like Microsoft have sophisticated certification processes, not only earning money, but creating a culture of dependency on them.

The cultural blindness may have little to do with computers or their perceived mystique. It may be a natural response to complexity: some people like using complexity as a subtle means of taking advantage over others less knowledgeable about the rules. For example, the UK tax system was too complex, and the Inland Revenue itself could not cope. Rather than simplify the system so that more people could cope with it, a law was passed effectively to require people to calculate their own tax. Thus nothing was simplified, but with the backing of the law, the country's population had to take it on themselves to be responsible for understanding an unnecessarily complex system.

Another example, backed by European directives, is just as computer technology would have permitted the introduction of simple user interfaces to car management systems, it has been outlawed. The reasoning is that only manufacturers should be able to adjust emission settings. Of course this is spurious; the consequence is that what could have been trivial for everyone has now become impossible for all but approved professionals.

Complexity suits manufacturers

Complex interactive systems are hard to understand. It follows that when a potential user is selecting between several systems (e.g., in a shop) they will be unable to make a rational choice based on usability and task-fitness. Instead, they will choose on brand name, appearance, price, and other superficial factors (such as colour). Thus, by making systems complex, manufacturers can push users into making decisions on much simpler factors, and purchasing patterns will become more predictable.

Which is better for a company: to sell a proportion of their systems, proportionally to distribution in the market, or to risk selling none because users can tell the system is inappropriate for some purposes? A cautious manufacturer might wish to hide possible problems — at least until the purchasing choice has been made.

In short, complexity — bad HCI — suits manufacturers, certainly for consumer products, and probably for products that are purchased in more formal procurement procedures (even if there are safety critical criteria).

Drama

Interactive systems, from video recorders to aeroplane flight decks, are highly complex, but they can be made to look deceptively simple by 'good' industrial design. The consequence is that problems only become apparent during a crisis. When a video recorder manual is lost and the programme to record starts *now*, or — more worryingly — when an aeroplane pilot is under pressure, mistakes are made. Often these mistakes can be traced to poor interaction design, including bad manuals, and poor quality control. Bad manuals, for example, are common because the technical authors are rarely in a position to adequately understand the systems. The problem is that computer systems can be made to look good enough for a demonstration or even to pass a test (Thimbleby, 1996), but without proper theories and knowledge of the precise nature of the systems, both informal and formal assessments reveal very little about the full range of possible behaviours of the systems. Indeed, consumer products often have cosmetic features and demonstration modes to entrance users, and hence delay the discovery of problems.

The Year 2000 bug

The Year 2000 bug is arguably the largest single problem ever facing humanity; it is certainly the most expensive problem of our own making! It arises from unprofessional practice by system manufacturers. The poor practice includes bad programming, bad design, bad diagnostic, bad documentation, and refusal to take responsibility.

The lack of care for users implicit in software warranties is well known (Thimbleby, 1990a), but the Y2k bug makes the culture of "usability is the user's problem" stark. Manufacturers charge customers for fixing problems, entirely of the manufacturer's own making. In some cases, this charging occurs several times as the manufacturers repeatedly upgrade their badly-engineered systems.

Insurance companies, the Halifax Building Society being a good example, refuse to insure against Y2k bugs. The Halifax's own contents policy ran from 1999 to 1900, until they noticed that they themselves were victim of the Y2k bug. But if they do not cover people against Y2k bugs, then the manufacturers are not going to be sued by insurance companies. Instead users have become responsible for fixing what — in the best tradition of oppression — they take to be their own personal problem.

Why is this behaviour: the ownership of problems by users, the denial of cover by insurers, the denial of responsibility by manufacturers, and the selling of training and upgrades to fix problems, confined to computer systems? Norman (1998) argues it is because the manufacturers have a 'teenage' mentality; and by keeping products before the threshold (Figure 1), they can stay juvenile as long as they wish.

HCI as user-centred

HCI problems are apparent by their effect on users, and the purpose of good HCI is to benefit the user. It seems self-evident that HCI should be user-centred. This is the central message of much work in HCI, notably Landauer (1995).

User-centredness is a forceful HCI slogan, but used uncritically it may impede HCI — this paper has given a list of reasons why users may not make sensible HCI decisions! For example, Nader exposed the misuse of "driver error" covering up responsibility for car accidents. Though some accidents are caused by drivers, not all of them are. Analogously, focusing on the user is important for HCI, but it is not the whole answer. Concentrating on user-oriented evaluation might lead, for example, to ways of camouflaging problems rather than avoiding them.

The Y2k problem is an example of poor quality in system programming. Arguably, many systems have poor programming behind them; thus the design quality of products is low and out of control. Therefore to improve HCI one has to concentrate on users and post-design problems. This is a practical response to the problem; but a strategic response — which should also come from HCI — would be to try to improve programming and design standards. For example, there could be usability certification of products, there could be certification of programmers, and so forth.

One of the problems, of course, is that most system engineering is awful, and programmers are just not skilled enough to make the design technically better (I talk as a computer scientist!) — thus, since designs *must* be made better, the onus is passed on to user-centred design to improve the design, who are able to make improvements (e.g., as Nielsen (1993) shows: "Forms are so difficult to fill in that, for one Australian insurance company, each claims form had an average of eight errors. The company staff had to spend over an hour sorting each one out." That is a problem that needs fixing, without programmers making it worse!)

Christensen (1997) is distinctly *not* user-centred. His argument is that users understand what they are doing not what they might be doing. Users work inside a value network, which may not be the best one for them as technology develops. We should also emphasise designer-centred design (Thimbleby, 1998b) since in fact it is designers who make new artefacts.

Culture affects experts

Professionals working in HCI itself are not immune to the unusability culture. It would be invidious to give examples of particular experts making mistakes (though I have my fair share!), rather Norman (1998) gives a very good summary of companies — such as Apple, Kodak, IBM — making major errors of judgement.

A typically British story, of being first yet ulitmately failing, is retold in a book poignantly called *User-Driven Innovation* (Caminer, Aris, Hermon & Land, 1996).

HCI as a natural science

Studying human computer interaction is itself a human activity, and therefore the very limitations that make HCI interesting are exactly the same limitations that make it a hard subject to advance!

Given, then, that HCI is too complex to study it is sensible not to take it *en masse*, but to take manageable parts. For example, we could take the computers as fixed, and study the human issues. Or we could take the human issues as fixed, and just study the computers. Psychology takes the former approach, computer science takes the second.

The conventional approach is for HCI to be treated much as a natural discipline. That is, the world is given, and HCI's job is to find out more about it. In turn this knowledge may help make future systems better. Compare this approach with psychology, which — apart from ethically dubious experiments — has no control over people and takes them as given. Or compare it with computer science, which as an applied area of mathematics, has a sort-of approach that assumes there is an 'ideal' computational model that in principle exists, as yet to be discovered.

To varying degrees, then, conventional HCI can be understood as a science. There is some truth "out there" (whether in people, society, or in a Platonic mathematics) waiting to be discovered. The methods of HCI generally reflect this underlying philosophical approach, that it is or aspires to be a natural science.

HCI lags behind commercial product development, which certainly raises new and interesting HCI issues. Product development is difficult and is the province of commercial organisations, usually requiring significant investment and resources. These facts in turn become an assumption: that commercial products are given (much like the natural world is given) and are therefore proper raw-material for much HCI research. Consider all the research on commercial word processors — compared with the paucity of research developing variations on word processors. Where is the experimental HCI that tries new designs, rather than tries new problems with existing designs?

So whilst many commercial products make interesting guinea pigs for HCI, there is less point in studying failures than in building better systems: HCI is not just a natural science …

HCI as an artificial science

In contrast to being viewed as a natural science HCI can also be understood as an *artificial* science (Simon, 1970). Nothing is given, because it is created by humans. Seen like this, HCI includes a design element, which creates new systems. Thus in contrast to the usual emphases of HCI, this view emphasises we construct new "truths," by building systems and by training users and requirements engineers … by changing the world.

This view of HCI is more proactive, but even so it is not without its problems. Constructive HCI is assumed to be either computing science or industrial design, and therefore is outside mainstream HCI. Computer scientists have enough trouble getting things to work, without worrying about HCI; and industrial designers make mock-ups that do not need to work before they are commissioned. Thus, whether HCI is a natural science or an artificial science, we still have bad user interfaces!

Thimbleby (1990b) has an entire chapter on the relation of science to HCI.

HCI lacks boundaries

At a more abstract level, HCI is not a mature discipline, with agreed boundaries.

Many sciences use computers for their own purposes. For example, computational chemistry is a discipline in its own right. Any such science inevitably involves human-computer interaction, and may indeed make specific contributions to HCI itself — for instance, through the discovery or analysis of new effects. Now, when the science is one that is anyway a 'component' discipline of HCI, such as psychology, it is easy to confuse doing it with doing HCI. Is a study of human perception HCI or psychology? Both? (The same might be said of work in computing that involves user interfaces — how can it escape them? — and therefore "is" HCI.) Of course, it does not matter much what it is, except that the confusion leads to the component discipline of HCI increasing its stake on HCI, and therefore seeing alternative views as competitive.

Or since computing science gets "user interfaces to work" it might appear from the computing blur that no HCI is necessary to get interfaces to work — it is all computing. Actually, computing gets interfaces to work, but not to work well; it is necessary but not sufficient.

And the lack of theory …

We've described a wide range of reasons why HCI does not get any better. Another reason is that is rather hard to do better without adequate theories.

When there are no theories, problems can be dismissed as coincidences, curiosities, or as "that's just how it is" (Lightman & Gingerich, 1992). Continental drift had been proposed as early as 1800 to explain the close fit of Africa's and South America's shapes, but the idea was dismissed — despite Wegener's data — until the theory of plate tectonics was developed in the 1960s. Both evidence and theory, a demonstrable explanation of the evidence, are required for the problems to be taken seriously.

Clearly HCI needs to develop theories and systems for the reliable design of complex technology, to clarify problems and make them visible at early points in the design process, where they can be managed, analysed and avoided. Unfortunately, HCI is very complex. A complete definition of the user interface of something as simple as a vending machine is at the limits of researchers. Researchers tend to concentrate on either idealised systems, or they gloss the difficulties in knowing what systems are. And without theories, product designers make arbitrary decisions, which in turn are hard to theorise!

HCI as a field is at a pre- or quasi-theoretic stage. Lots of 'small' theories and ideas are available, but industrial practice lags behind because current research does not scale up, nor is it packaged in a way that relates to the needs of designers. Research has polarised into a 'creative' end, proposing new techniques and applications, and an 'evaluation' end that assesses the impact or success of given systems. Proper evaluation is hard to do: commercial systems are poorly specified and what is evaluated cannot be precisely defined. Evaluation and other "outside-in approaches" rarely provide constructive insights that reliably can be fed into new designs, and in any case, designers feel successful enough not to need them — hence perpetuating inferior design.

In the 1980s, computer scientists were often lead designers, but as computers became more accessible and more pervasive, changes occurred: programming, as opposed to graphic designing, became much harder (the programmer-year effort to compete with commercial products increased dramatically); and evaluative approaches (legitimately) took a dominant position in the field of HCI. The result of this has been a down-play in the contribution of theory to design.

Everyone is an expert

It is very hard if not impossible to work within HCI without having a personal stake in the field. Some of us use Macintoshes, some of us use Unix, some PCs. Whichever we use, we live in a community of users of the same sort of system — many things we do reinforce the wisdom of being in this community. Other people in the same community are more helpful (of course!) and they have solutions we can covet to help us, whereas people working with other systems do not know the answers to our personal problems, and they are not using solutions that we would covet. There are even lively magazines and other fora that emphasise the value of each system to its own community — and often in contrast to the comparative disadvantage of the alternatives.

Almost certainly, then, anyone who finds a way of improving one of these systems is not going to impress the rest of the HCI community!

Likewise, and on a wider scale, we all have video recorders, mobile phones, calculators, and have everyday experience of various sorts of gadgets. Often, then, we have a stake in a particular solution. Other sorts of solution are of no personal interest, because they do not help us use the particular gadgets we own.

Solutions are hard to publish

HCI as a conventional academic discipline progresses by shared use of refereed literature. Quality work is published, and what is published tends to define the current trends in HCI.

Unfortunately, one of the damaging consequences of the fast-paced development of technology is that for any focused development in HCI, there is almost certainly a commercial product that excels some aspect of that development. Now, since any referee knowing this will have a stake in that technology, they are likely to deprecate the contribution. Thus there is a tendency for the HCI literature to concentrate on procedures and users, rather than new technical solutions. In other words, the literature does not look at alternatives to commercial products; indeed, explicit discussion of products is deprecated as commercial “product reviews.”

In short, academic HCI has little relevance to product introduction, because the publishing process tends to avoid such issues — for the reasons given above, and for the more explicit reasons of commercial reality: academics neither want to libel nor advertise particular manufacturers as it is not professional to do so.

What can be done?

Manufacturers will not improve their products when their products are consumed by an uncritical public, and the public will remain uncritical while there are no standards on which to base effective criticism.

Public understanding of science

HCI should include an explicit public understanding of science element. This is valuable in its own right, but is also because HCI should have a vision about changing culture — inspiring, particularly, children to go on to identify and fix usability problems. A spin off would be a greater appreciation of the way in which our society depends on deeper concepts from computing science: it doesn’t just depend on the boxes called computers, but depends on programs (and documentation) being well-designed and appropriate for their tasks.

HCI needs more books like Don Norman’s classic *The Psychology of Everyday Things* (1988) — renamed *The Design of Everyday Things* for the paperback), and Bell, Witten and Fellows’ *Computer Science Unplugged* (1999); however most books, to date, have been “hindsight” or social criticism. HCI needs more constructive approaches: popular books emphasising the deep relevance of HCI design to everyday life, to help consumers be more discerning, and to help designers be more likely to succeed with *new* products.

A practical way forward

There are many ways forward, particularly engaging human approaches — addressing political, human factors and other issues head-on.

Another, perhaps more systematic, way of demonstrating the effectiveness and value of research in HCI would be to build a demonstrator system, a sort-of laboratory bench. We will call such a system an Interactive Design Assistant, or IDA. This idea fits ‘conventional’ HCI; it is recommended by Gould, Boies and Lewis (1991) as one of their four design principles — to integrate design, so that all aspects of usability (user interface, help systems, training, documentation) should evolve concurrently.

An IDA would not just be a simulator, but would have other features. Given a common representation (such as graphics and program), all the following are possible, and can be achieved in an integrated way, consistent with the ‘central’ program specifying the user interface:

- Simulate user interfaces.
- Provide various usability ratings.
- Generate user manuals.
- Analyse theoretical performance.
- Log and evaluate actual use.
- Animate demonstrations.
- Generate hardware designs.
- … and even arbitrate in usability competitions!

Relating to the public understanding of science theme: an IDA would be able to demonstrate some of the fundamental ideas, such as the wide application of HCI in everyday life, and would make an ideal web site or CD to be associated with such a book!

An IDA would operate on the World Wide Web (using a browser-like interface) and hence make HCI research very widely accessible. It will enable widespread uptake and severe testing of the ideas, as well as far more effective collaboration and support of related research. Other researchers will be able to contribute designs and make them available for critique or as exemplars of particular ideas. An Assistant will automate, and hence make reliable and efficient, many aspects of design, such as quality control and evaluation, and will provide links between all parts of the design process. An IDA would lend itself powerfully to use on the web, such as annotating interface bugs, or presenting usage statistics in novel ways in documentation.

Most designers adopt an appearance-oriented approach, and there are many powerful tools to do this that are very good at visual simulation, but which are (and unfortunately have to be) programmed in an *ad hoc* fashion (Narayanan & Hegarty, 1998; Sharp, 1998). So an important aspect is to package the theory into an easy-to-use IDA to make theoretically-sound methods available and acceptable to the design culture; conversely, the ability to undertake real, large-scale and complete design projects, with evaluation, would challenge the current theories and, in turn, lead to many new insights and developments.

As Shneiderman says (1998), "the advantages of specialised user-interface software tools for designers and software engineers are large. They include an order-of-magnitude increase in productivity, shorter development schedules, support for expert reviews and usability testing, ease in making changes and ensuring consistency, better management control, and reduced training necessary for designers." These are attainable goals for an IDA.

While user interface design remains atheoretic (from a computing perspective), there will be no pressure on anyone to improve. HCI research is based on the claim that considerable improvements are possible, but that the improvements have to be well-founded, integrated, and themselves easy to use. Once packaged and widely accessible, then the evaluation of systems will become much easier. One might anticipate consumers testing and comparing products with systems such as those proposed, and hence putting greater pressure on manufacturers to design good products, rather than — as they do at present — merely attractive ones, whose usability problems are disguised from purchasers at the time of sale. An IDA, as proposed, would represent a qualitative leverage in design approaches. An IDA would ultimately not depend on any one research individual, group or organisation, and the scope for wider, autonomous, collaboration would be greatly extended.

An IDA would readily support thorough system reviews, somewhat as Dijkstra longed for in his 1972 Turing Award lecture (Dijkstra, 1987). It would also support system design using intellectually manageable approaches, another of Dijkstra's core ideas. As Dijkstra points out, such constructive approaches increase confidence in designs; in contrast, empirical testing, on which so much HCI currently depends (because there is nothing much better) can only show the presence of bugs — and it has driven HCI into a user-centred problem-oriented discipline (see Landauer (1995) for arguments to this effect) to the loss of designers, who could use an IDA.

Beneficial impact

Why would an IDA help improve HCI?

To be adopted an IDA would have to improve business processes; it would have to speed the time from design to product, or it would have to speed the design iteration cycle (supposing iterative design is being used). Whether an IDA would speed the design process is a matter for its own design to determine — is it fit for purpose? The scheme proposed here is that the IDA would have an open-architecture and run on the Web. It would therefore become a community project, rather like Linux perhaps, and by harnessing such co-operation could likely achieve the productivity gains required.

An IDA would have to improve the market of (good) interactive systems. Businesses need to stay in the market to reap the benefit of better HCI, and this presupposes they are selling products. Here an IDA could help by creating a 'shop front' where users could try products on the web. It would be possible for users (or consumer groups representing users) to program benchmark task sequences: thus a user could select a suite of tasks, and see them run on a variety of designs. Users would then be able to select systems that best suited their needs.

Finally, an IDA would have to reduce after-sales costs. By improving usability, by improving manuals, and by helping provide better task-fit, an IDA should help reduce the numbers of users who buy the wrong products, or who buy products they do not fully understand. Thus, an IDA would show its success in the long term, as companies reduce the level of after-sales support. In turn, this would allow companies to invest in better current designs.

Conclusions

This paper has made three sorts of claim, and each sort of claim can and should be tested:

1. *A range of "cultural" claims.* Are these claims correct? Where is the evidence? What are the confounding factors, for example which would enable the cycle of bad usability to be broken?
2. *A proposal that public understanding of science will help.* Despite its popularity, usability seems not to be treated in the public arena except by journalism, and then usually to celebrate new technologies and new features. There is a considerable scope to exploit the public's interest in consumer gadgets to promote usability.
3. *A specific proposal for a design tool.* Such a tool remains to be built! A companion paper, also submitted to this conference, describes such a tool in more detail.

Acknowledgements

Ann Blandford, Gary Marsden and Prue Thimbleby made many helpful suggestions.

References

T. BELL, I. H. WITTEN & M. FELLOWS (1999) *Computer Science Unplugged*,. See http://unplugged.canterbury.ac.nz/

D. T. CAMINER, J. ARIS, P. HERMON & F. LAND (1996) *User-Driven Innovation* , McGraw-Hill

C. M. CHRISTENSEN (1997) *The Innovator's Dilemma: When Technologies Cause Great Firms to Fail*, Harvard Business School Press.

E. W. DIJKSTRA (1987) "The Humble Programmer," *ACM Turing Award Lectures*, 17–32, Addison-Wesley.

W. W. GIBBS (1991) "Taking Computers to Task," *Scientific American*, **277**(1): 64–71.

J. D. GOULD, S J. BOIES & C. H. LEWIS (1991) "Designing for Usability — Key Principles and what Designers Think," *Communications of the ACM*, **34**(1): 74–85.

T. LANDAUER (1995) *The Trouble with Computers*, MIT Press.

B. LAUREL (1991) *Computers as Theatre*, Addison-Wesley.

A. LIGHTMAN & O. GINGERICH (1992) "When Do Anomalies Begin?" *Science*, **255**(5045): 690–695.

D. MACKENZIE (1994) "Computer-related Accidental Death: An Empirical Investigation," *Science and Public Policy*, **21**(4): 233–248.

N. H. NARAYANAN & M. HEGARTY (1998) "On Designing Comprehensible Interactive Hypermedia Manuals," *International Journal of Human-Computer Studies*, **48**(2): 267–301.

D. A. NORMAN (1988) *The Psychology of Everyday Things*, Basic Books, Inc..

D. A. NORMAN (1998) *The Invisible Computer*, MIT Press.

J. NIELSEN(1993) *Usability Engineering*, Academic Press.

J. A. PAULOS (1990) *Innumeracy*, Vintage.

M. PIATELLI-PALMARINI (1994) *Inevitable Illusions*, John Wiley & Sons.

J. SHARP (1998) *Interaction Design for Electronic Products using Virtual Simulations*, PhD thesis, Brunel University.

B. SHNEIDERMAN(1998) *Designing the User Interface*, 3rd ed., Addison-Wesley.

H. A. SIMON (1970) *The Sciences of the Artificial*, MIT Press.

E. TENNER (1996) *Why Things Bite Back*, Fourth Estate.

H. THIMBLEBY (1990a) "You're Right About the Cure: Don't Do That," *Interacting with Computers*, **2**(1): 8–25.

H. THIMBLEBY (1990b) *User Interface Design*, ACM Press Frontier Series, Addison-Wesley.

H. THIMBLEBY (1992) "The Frustrations of a Pushbutton World," *1993 Encyclopedia Britannica Yearbook of Science and the Future*, 202–219, Encyclopedia Britannica Inc.

H. THIMBLEBY (1993) "Computer Literacy and Usability Standards?" *User Needs in Information Technology Standards*, 223–230, C. D. Evans, B. L. Meek & R. S. Walker, eds., Butterworth-Heinemann.

H. THIMBLEBY (1996) "Internet, Discourse and Interaction Potential," in L. K. Yong, L. Herman, Y. K. Leung & J. Moyes, eds., *First Asia Pacific Conference on Human Computer Interaction*, 3–18.

H. THIMBLEBY (1998a) "The Detection and Elimination of Spurious Complexity," *Proceedings of Workshop on User Interfaces for Theorem Provers*, R. C. Backhouse, ed., Report 98–08, 15–22, Eindhoven University of Technology.

H. THIMBLEBY (1998b) "Design Aloud: A Designer-Centred Design (DCD) Method," *HCI Letters*, **1**(1): 45–50.

Biography

Harold Thimbleby is Professor of Computing Research at Middlesex University, London.

Harold gained his PhD in 1981 in Human-Computer Interaction, and his research since then has been concerned with making complex devices easier to use. He is a computer scientist, with a passion for user centred design and analytic approaches to design. He wrote *User Interface Design*, published by Addison-Wesley in 1990, and has 192 other publications.

In 1987 he was awarded the British Computer Society Wilkes Medal. He is on the British Engineering and Physical Sciences Research Council's Computing College, and on many editorial boards, including *Personal Technologies* and *International Journal of Human Computer Studies*. He is on the International Federation of Information Processing (IFIP) committee on computers and ethics. He has held visiting posts in Australia, Canada, New Zealand and Switzerland, as well as in industry.

His web site is at http://www.cs.mdx.ac.uk/harold

Various papers are available at http://www.cs.mdx.ac.uk/harold/srf

With Penny Duquenoy (a PhD student) he is presenting a paper at Interact'99: "Justice and Design."

Making user-centred design usable

Richard Whitehand (richard.whitehand@nomos.se) & Nigel Claridge (nigel.claridge@nomos.se)
Nomos Management AB, Box 119 (Svärdvägen 3A), S-182 12 Danderyd, Sweden
http://www.nomos.se

Nomos is a usability consultancy, and as such we do *not* provide a full interface design and application implementation service to clients. Thus the way we work and the outputs from our work have to be usable for interface designers and systems developers – we to a large extent act as a bridge between users and developers.

As we are actively involved in ergonomics and usability standardisation work, we make significant use of these standards in our work (particularly ISO 9241 and 13407). We have a range of tools and methods that we use throughout the different phases of user-centred design as described in ISO/FDIS 13407.

Our approach to user-centred design and the methods we use are outlined in the Appendix at the end of this document. Two particular methods we have used in several projects and feel are worth discussing in a little more detail (with regard to 'making user-centred design usable' for interface designers and developers), are: ***"usage scenarios"*** (scoped out in the later stages of context and requirements definition); and ***"parallel design"*** (as a preliminary activity in interface design work). These are outlined and illustrated with examples from current work with the design of a new Web site for the Swedish Tourist Board.

Usage scenarios

In many object-oriented systems development models, developers document "use cases". These are often very functionally oriented, documented in detail, and plentiful in number. Usage scenarios, however, are not use cases, but simple realistic examples of users and their goals with using a system – they are, by nature, task-oriented.

Whilst a clear and concise documentation of the 'context of use' and 'user and organisational requirements' (as they are called in ISO/FDIS 13407) are useful, these are often insufficient by themselves to transfer a real understanding of users, their situation and their goals, to the designer. The purpose of "usage scenarios" is to illustrate 'real life' examples of users and what they would like to do. They are not written for all situations or tasks, but for a selection of typical/representative ones and can act as a 'high level' illustration to use cases or task analyses.

On the following page are some examples of scenarios, which were documented for the Swedish Tourist Board as part of the work on their new tourist web site for Sweden. The purpose of this Web site is firstly to attract prospective tourists to Sweden and to provide those people who have decided to come to Sweden with information about what to do, where to stay, etc – to help sell the Swedish tourist industry.

Tourist 1	Tourist 2	Business traveller 1
Dave Allen and his two friends live in Ireland and have decided to visit Sweden on a golfing holiday in August. They want to play at four different courses (one of which is to be Barsebeck as they have heard that this is a fine course), all which should be in easy driving distance of each other. Dave will need information about each of the courses in printed form for his friends to look at. They want to stay at a nice hotel (not too expensive) with good food and that is also close to the courses. Dave is also interested to know if there are any other activities going on at that time in the vicinity. They plan to fly to Sweden and then rent a car.	Michelle Monet lives in Paris and plans to visit Göteborg for a two-week vacation. She wants information about hotel accommodation in Göteborg, restaurants and night clubs, museums and theatre, and sailing in the archipelago. She wants to know what other attractions are available in the area (she will rent a car if necessary) and also what is on during that two week period.	Nigel Bevan is a businessman living in London. He is being sent by his company to negotiate a business deal with Nomos in Stockholm in the month of February. He expects to be in Stockholm for three week days (two nights). He will be staying at the Sergel Plaza Hotel and wants to know more information about the hotel. He is not familiar with Stockholm and will need to know how to get from the hotel to Nomos in Mörby (quickest and cheapest). Nigel Bevan has heard that there is a fine performance of Figaro at the Opera and would like to know if there are tickets and if the Opera is in walking distance from the hotel. After the Opera he would like to go to a traditional Swedish restaurant. On his other evening he would like to invite Nomos to dinner at a good Spanish restaurant renowned for its good wine and basque food. As he is flying to Stockholm, he would like to know how to get from Arlanda airport to the city centre and what sort of clothing to bring.

Such scenarios help convey a ‘down to Earth’ understanding of the user’s situation and what they want to do (typically 2 or 3 per user group). They can easily be used to ‘walkthrough’ design ideas as designers think through and sketch out their initial ideas. Later in the design process they can be used to test different design concepts. Tasks inherent in the scenarios can form the basis of more formal usability testing later in the design process. (It is of course important to remember that these are only examples and not a complete reflection of all requirements for the interface).

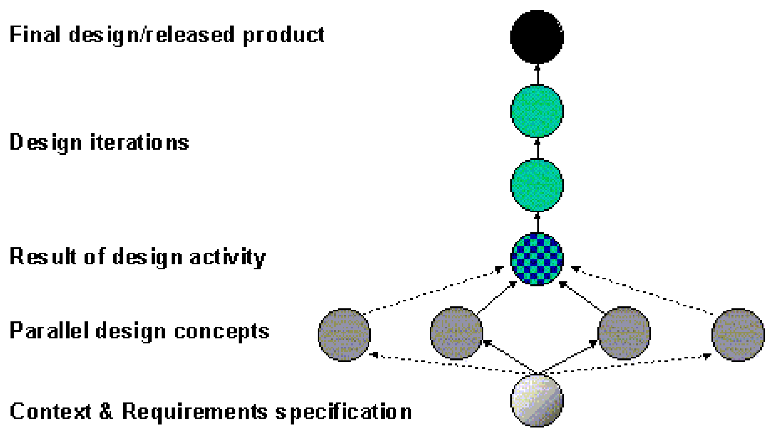
Parallel design

Parallel design is very useful in enabling multiple interface design concepts to be generated and considered before choosing the most appropriate one. It helps avoid the problems of designers either settling too quickly for a design concept before trying out alternative ideas, or later conflicts between designers who believe in different solutions to the problem.

Supported by the typical scenarios of use and knowledge of the context/ requirements, two or more UI design teams (typically 2 people in each) are given the assignment of independently producing design concept sketches. No more than two man-days per group are spent on this activity and the outcome is a set of rough sketches of how the main content/functionality of the product should be implemented.

The teams then meet in a workshop, together with user representatives and other stakeholders in the development. Different solutions are discussed and ‘walked through’ with the aid of the scenarios. The aim of the workshop is to decide upon one overall conceptual design for continued development (this may be one of the solutions, or a solution arrived at as a result of discussing the pros and cons of the different interface sketches). This concept is then iteratively developed and tested together with representative users.

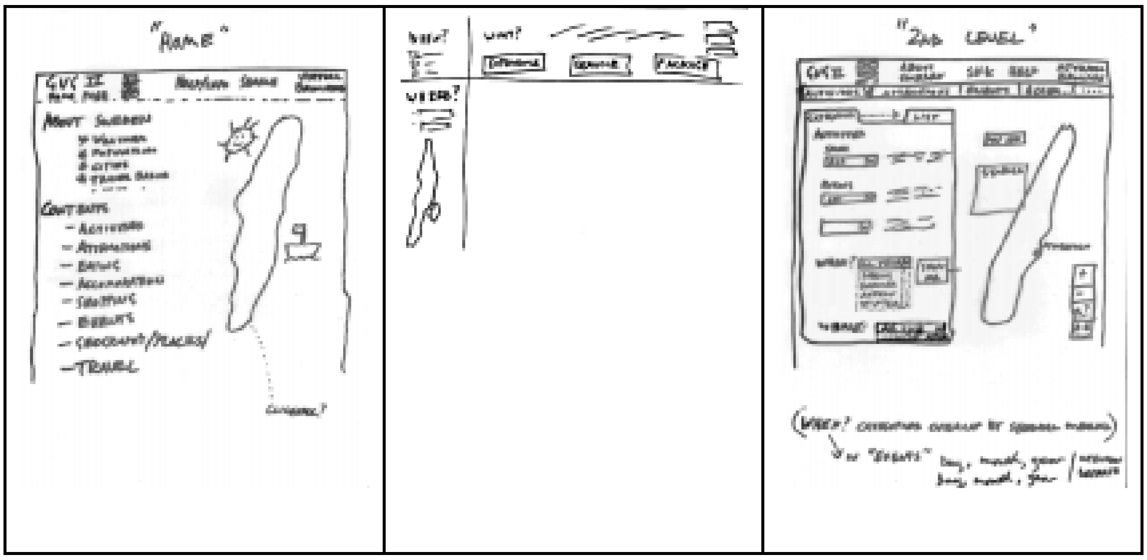
The focus of this early design work is on considering the overall function and flow of the interface in supporting user’s tasks, rather than graphical appearance issues.


Final design/released product
Design iterations
Result of design activity
Parallel design concepts
Context & Requirements specification

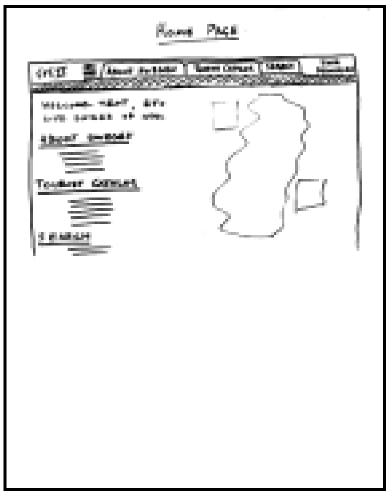
Parallel Design (Ref: J.Nielsen, Usability Engineering, 1994)

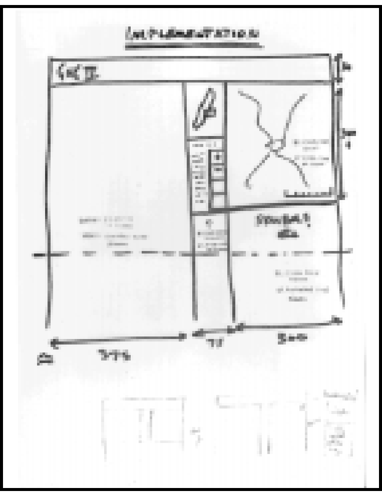
In the case of the Web site for the Swedish Tourist Board, 4 groups interface design groups (2 people in each) were used, and some of the sketches from the parallel design activity can be seen below.

Parallel design concepts (examples):

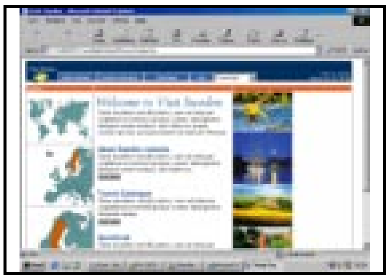
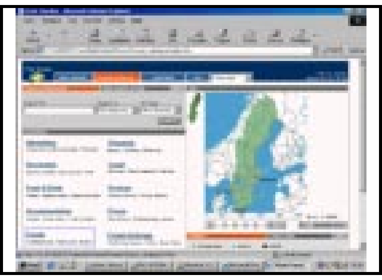
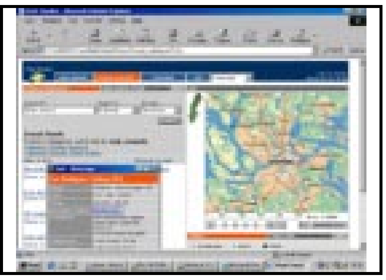


Post-parallel design workshop iteration (examples):





Later iteration, now in a browser window (examples):

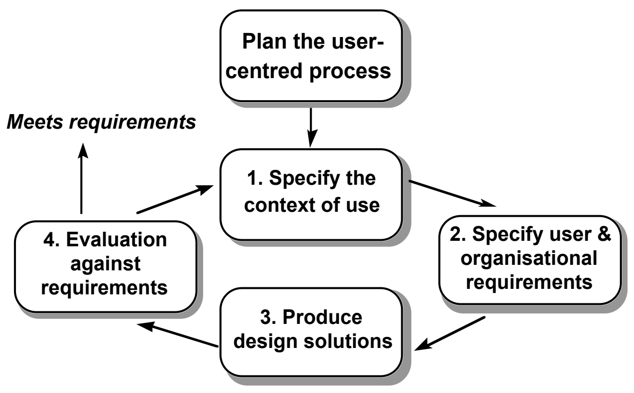




Summary

We feel that “usage scenarios” and “parallel design”, appropriately supported by other user-centred activities, are two particularly useful and practical methods that can help make user-centred design more usable for user interface designers and developers. Both can be efficiently employed early in the development process to help bridge the user-centred requirements and user interface design work.

Appendix

ISO 13407 : Human-centred design processes for interactive systems


Plan the user-centred process
Meets requirements
1. Specify the context of use
2. Specify user & organisational requirements
3. Produce design solutions
4. Evaluation against requirements

Our approach and the methods we use		
	Context analysis 1 *(High level – half day workshop with all key stakeholders in the design)*	• Overall project goals and sub-goals • Brief product description • Key product characteristics • Project limitations / dependencies • User groups (prioritised) • User group characteristics (e.g. age, duration/frequency of use, skills, culture, handicaps, computer experience, web experience) • Envisaged user tasks
	Requirements gathering, e.g. User interviews Surveys Focus groups	Activities here vary significantly depending upon the nature of the product and it's users. Work here concentrates primarily on collecting information about *WHAT* the users want to see/do.
	Review existing designs: Usability test current product Obtain input from helpdesk /support staff for the product Review competitors and functionally related/similar products	These activities are a means of obtaining usability/user interface design knowledge about existing interface implementations. This can provide useful input to *HOW* (and how not) to implement future interfaces.
Pre-design	**Context analysis 2** *(Review of 1 & supplementation)*	In the light of requirements gathering work, and prior to starting design work, a review of the context of use is carried out. This focuses more on user tasks and sub-tasks than the first context of use and includes the formation of some typical usage scenarios (see below).

	Scenarios of use	A series of brief scenarios of use are written – where different groups of users exist then several may be written for each group. Each is a short 'real life' descriptions of the user and a goal (or set of related goals) they have in using the system. Written in plain English, these are typically 2 or 3 short paragraphs of text in size.
Early design	**Parallel design**	Supported by the typical scenarios of use and knowledge of the context/requirements, 2 or more UI design teams (typically 2 people in each) independently produce design concept sketches. The teams then meet in a workshop, together with user representatives and other stakeholders in the development. The aim of the workshop is to decide upon a single overall conceptual design for continued development.
	Concept iteration	The concept resulting from the parallel design activity is iteratively developed, initially on paper and then other mediums (e.g. HTML / Graphics package, Visual Basic, etc).
	Combined expert and user testing	During the iteration of the chosen concept, usability experts review designs and carry out 'participatory' evaluations together with users (these are task-related evaluations, carried out in a less formal manner than a traditional usability lab test, and where the expert can 'fill in' missing interaction/flow in the interface so that tasks can be simulated in some way).
Late design	**Usability lab testing**	When a functional prototype is available then more formal usability tests can be carried out. The overall goal of such tests is usually to uncover particular interface design issues that cause user difficulty, but for work-related interfaces it can also be to measure usability in terms of user efficiency and effectiveness in carry out tasks.
	Subjective questionnaires (SUMI / WAMMI)	Standardised user satisfaction questionnaires can be used to gather feedback as designs are iteratively improved later in the development process. Especially useful when the user group is not immediately accessible.
After release	**Usage statistics**	Can provide useful information about usage patterns that may point to interface issues that could be improved.
	Field observation	How is the product actually being used in 'real life' and how can the next version be better?
	Subjective questionnaires (SUMI / WAMMI)	(As above) – Allow comparison/benchmarking with software or web interfaces in general across several different usability scales, indicating aspects of usability that may be in need of attention.

Biographies

Richard Whitehand trained as an ergonomist at Loughborough University of Technology (UK), completing a four-year Honours degree in Information Technology and Human Factors in 1995. Since then he has been working as a consultant usability specialist for Nomos Management AB, a human factors consultancy in Stockholm, Sweden.

His consultancy work has mainly focused on supporting the design of user interfaces from a usability perspective (recently with a focus on interactive web-based applications such as Internet banking, travel and employment services). He has supported the development of a wide variety of software applications and Web sites for Swedish and international organisations – working with designers and developers both at the early stages of requirements analysis and conceptual design, through to prototypes and fully implemented systems.

Nigel Claridge is Senior Partner and co-founder of Nomos Management AB. His specialist areas are the integration of user-centred design principles and activities in to product and systems development processes, in particular usability process improvement, user interface design and usability evaluation. He has worked as a consultant to a wide range of multinational companies and held tutorials and seminars on user-centred design both in Sweden and abroad.

His vision is the natural integration of usability activities in development processes with the aim to create systems and products that are easy to use and meet the needs of intended users. He believes that this will be achieved mainly by decision-makers through to developers improving their knowledge and understanding of usability and the benefits of user-centred design.

Nomos Management AB is an independent usability and human factors consultancy that supports organisations develop and introduce products and systems which are effective and easy to use. Nomos is Microsoft’s usability partner and a European Usability Support Centre. Amongst our clients are ABB, Enator, Ericsson, Föreningssparbanken, Handelsbanken, Investor, Microsoft, Nokia, Nordbanken, SAS, Stockholm’s Stock Exchange, Scandinavian Online, S.E.Banken, Scania, Skandia, Swedish Labour Board, Swedish Defence, Swedish Post Office, Swedish Tourist Board, Telia and Yellow Pages.