Usability and Design Decisions in Software Development

Inger Boivie

Department of Information Technology
Uppsala University
Box 337, SE-751 05 Uppsala, Sweden

Technical report 2001-021
Oktober 2001
ISSN 1404-3203

Usability and Design Decisions in Software Development

Inger Boivie
Department of IT, HCI
Uppsala University
Uppsala, Sweden
Inger.boivie@hci.uu.se

Contents

Abstract

Despite the attention that has been paid to usability in the last few years, the world is still full of inadequate software and frustrated users. The aim of this study was to deepen the understanding of how usability matters are handled in software development projects, particularly at the Swedish National Tax Board (Riksskatteverket – RSV) and the Swedish National Board for Social Securities (Riksförsäkringsverket – RFV).

The main focus of the study was usability and interaction design decisions in software development projects, that is, considerations in the software development process that have implications on the usability of the resulting system. Who makes the decisions about the interaction design? When are they made? What are the sources of input and the main constraints? What support regarding usability matters does the decision maker have access to?

The study was conducted by means of a series of semi-structured interviews with eight developers and usability people in the two organisations.

The study shows that there is no simple answer to who makes the interaction design decisions and when they are made. Rather, it is a case of everybody (developers , project managers, client representatives and user representatives) making this kind of decisions, all the time. Many decisions are never made, they just happen as a result of somebody coding a bit, or modelling a bit. The responsibility for the interaction design was unclear in these organisations, causing frustration. Use cases were the basis of all design, and thus crucial to the usability of the resulting system. But, use cases were reported to be difficult write. They easily turned into “system operations” providing little or no support for interaction design.

One conclusion is that usability requires hands-on activities throughout the project, or it “gets lost”, this is particularly important during construction. One way of achieving continuous attention could be to incorporate a usability role in the system development process and give it sufficient status.

Introduction

Un-usability is everywhere

The world is full of inadequate and difficult-to-use software and frustrated users. Software that frustrates our attempts to, for instance, perform our work tasks or search for information on the web.

Then, why does the software industry develop systems, web sites and products with poor usability? Why do not users get easy to use and easy to learn software that is pleasant to use, fulfils their needs and makes them happy, or at least not frustrated?

According to Clegg (1997) usability issues are on the agenda in most IT development projects. The respondents in their study estimate that as many as 60-70 % of the projects addressed usability matters “successfully”. They go on to say, however, that there is criticism that the area is still not sufficiently understood, and that the view of usability is rather mechanistic. Moreover, the respondents reported that users are rarely successfully involved in the projects, and that actual levels of user involvement are low. This seems to contradict the claims that usability is successfully addressed in the majority of development projects. Gould,

Boies and Ukelson (1997) point out that involving the users is imperative to usability. In addition, many software development projects adopt other development models, for instance, Rational Unified Process™ (RUP), which is architecture-centred rather than user-centred as described by Kruchten (1998).

Then, what happens to usability in a typical software project? Where does usability come into the picture? How is usability considered and by whom? Is usability reduced to "icing the cake" and designing cool graphics or does it inform all decisions that in one way or another affect the interaction throughout the project?

The purpose of this study was to explore some of these matters in order to deepen the understanding of what happens to usability in a software development project. Usability depends, to a large extent, on the design and functionality of the system or product. Thus questions about who makes the decisions in these matters seemed one way of exploring what happens to usability. Who, for instance, makes the decisions about what services to include in the system and what constraints should be placed on these services? Who decides on the presentation of the services to the user? And who decides on what information should be available to the user? When are these decision made? What support regarding usability matters is provided? What input does the person making the decision have and what are the constraints?

Scope

Software development is an extremely complicated process, where usability is one aspect out of many that must be taken into consideration. Usability activities in software development projects typically include user and task analysis, prototyping and evaluations. Usability is, however, a result of good design – analysis and evaluation activities can only serve as a basis for design decisions. Therefore, the focus of this paper is *design* and *usability*.

Usability issues in software development projects

ISO 9241-11 (1998) defines usability as

> *"The extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use."*

In other words, usability means that the services and information provided by the system suits the goals of the user and that the user can perform his/her tasks in an efficient and satisfactory way.

Usability issues are of course not just a matter of the software development project. They are equally a matter for the user organisation as a whole, for instance, in the decision on what technical platform to use in the organisation or whether software development should be in-house or outsourced.

The focus of this study is, however, software development projects. The term "software development project" is vague and equivocal – in this paper the term is used in a quite broad sense, as specified below:
A software development project is limited in time and in size. It starts with some kind of "order" for a particular piece of software. The order is more or less formal in its nature. The project is finished when the software has been delivered and accepted by the user organisation, or the project may be abandoned during the development process. The project includes the development team (and all the supporting workers, such as project managers,

configuration managers, etc) and the people involved from the client side, for instance, user representatives and domain experts and/or marketing department people.

In a software development project, a great number of decisions are made from the very start, to the very end of the project. These decisions vary greatly as to their type and significance, but many of them have one thing in common, they affect the usability of the resulting system. Some decisions are to do with the actual system whereas others are more to do with the organising of the project work, as described below.

Matters regarding the system, e.g.	Project organisation matters, e.g.
• What development platform to use • How to design the conceptual model • What to include in a particular release • What controls to use in a particular dialogue	• What software development model to use • When and how to involve users, if at all • The number of iterations and what stop criteria to apply • Who should design and implement a particular service or use case

Decisions are made on all levels, from overall decisions that affect the whole system or project, to decisions about details in the user interface or work division in the team.

- Macro level decisions
  for instance, what development platform to use, how to design the system architecture, how to organise the project, whether or not to include usability expertise, what software development model to use and how to involve the users
- "System overview" level
  for instance, the design of the conceptual model and what services to include (in a particular release)
- Detail level
  for instance, how to design and implement a particular service or use case, i.e. how to split it up into a number of steps or dialogues and what constraints to place on it
- Micro level
  on the micro level the individual designer or developer chooses what control to use for a particular instance of interaction and how to implement it, what names to use for the fields and buttons in the user interface, etc.

Naturally, a majority of these decisions are design matters - design of use case models, class diagrams, databases, prototypes, conceptual models, dialogues, etc. The question is, how many of these decisions are informed by usability, and in what ways?

Usability and design

How then, does the designer move from usability aspects to design? It is far from self-evident, how, for instance, effectiveness is built into an application, or how the interaction is designed for efficiency. Van Welie, van der Veer and Eliëns (1999) propose a layered model for breaking down usability into aspects that are possible to incorporate into a design. The model contains four layers – starting with the three usability aspects as described in the ISO definition, *effectiveness*, *efficiency* and *satisfaction*, at the top level. The next level contains *usage indicators* that can be observed and measured in practice when users use the application, for instance, learnability, satisfaction and memorability. These indicators are

affected by *means*, such as consistency and task conformance. The bottom level contains the *knowledge domains* required for good design. These are user model, design knowledge and task model.

This model helps making the shift from aspects in the design to usability aspects. Usability requires that design decisions are informed by knowledge from the knowledge domains and that the designer knows how to apply the means as specified in the model. Ideally, the person making the design decision has the knowledge that helps him/her apply the correct means in order to achieve the usage indicator/s that will produce the desired effect. Is this usually the case?

Interaction versus user interface

As pointed out above, usability encompasses much more than just the design of superficial characteristics of the user interface, such as colours, fonts and icons. In order to avoid this "cake-icing" approach to usability, the term *interaction design* is used instead of user interface design in this paper. Below is a definition of interaction and what that implies for the interaction design that may help the reader understand the concept.

According to Dix, Finlay, Abowd and Beale (1998) human-computer interaction (HCI) can be defined as
"any communication between a user and a computer, be it direct or indirect. Direct interaction involves a dialog with feedback and control throughout performance of the task. Indirect interaction may involve background or batch processing." (p. 3)
Dix, et al, also define an underlying basis of their view of HCI, namely that people use computers to accomplish goals within a particular domain. They do this by means of performing tasks, i.e. operations that manipulate the concepts within that domain.

Thus, the interaction as such is related to a particular goal and task. People do not move about in user interfaces aimlessly. Human-computer interaction is mediated by but *not* limited to the user interface. It includes:

- *What* the user can do – i.e. what services the system offers and what constraints are placed on these services and to what extent the user can fulfil his/her goals
- *How* the user can perform his/her task – i.e. in what order he/she can perform the steps contained within the task, as well as how the user interacts with the controls of the interface
- What *information* is available to the user during the performance of the task

Thus (human-computer) interaction design encompasses any design that in one way or another affects one or more of the above items. Interaction design is not just about painting on an appropriate user interface. In user interface design some of the main concerns are getting the metaphors right, grouping and placing the information in the best possible way and choosing the best controls for input and output. These matters cover the "how" part of the user interaction. To design the "what" part of user interaction and the "what information" part, we need to look at all parts of the system. In the three-tier system architecture, the different parts are defined as the presentation (user interface) tier, the business logic tier and the information tier as described by Allen and Frost (1998). This means that human-computer interaction is not only defined by and controlled by what goes on in the presentation tier, but also by the business logic tier and the information tier.

The design process and design decisions

Design problems are typically "wicked problems" as defined by Rittel and Webber (1973) in that they

"…defy efforts to delineate their boundaries and to identify their causes, and thus to expose their problematic nature."

This means, for instance, that understanding the problem is the same thing as solving it. The problem cannot first be understood and then solved, since the solution determines what information is needed to understand problem. Another characteristic of wicked problems is that there is no knowing when you are done solving it. Stopping rules are external, such as time or money or endurance. Nor are there any true or false solutions, there are only good ones and bad ones and the number of possible solutions is infinite.

A good design process must allow for the wickedness of design problems. Löwgren and Stolterman (1998) list some aspects that are characteristics for such a design process. They point out, for instance, that the design solution must be created in parallel with the identification of the problem. The goal of such a design process is to create a solution that is good enough, but nevertheless just one out of an infinite number of solutions. All design is context-dependent, and can only be evaluated within a context. This means that usability, for one thing, cannot be evaluated against a set of general criteria. The designer continuously shifts his/her focus from detail to overall picture and back, from abstract to the concrete and back. The process is neither linear nor iterative, but completely dynamic and reciprocal.

Gulliksen and Göransson (2000) point out that one distinguishing trait of interaction design is that decisions are often not explicitly made or even intentional. Design just *happens* as a result of somebody doing something – writing a bit of code or doing a bit of modelling. Thus, traditional decision making theory describing a decision as the final choice between alternatives, preceded by a judgement phase, is not applicable to design decisions. Another approach to decision making is required for understanding design decisions. Naturalistic decision making (NDM) provides one such approach, as defined by Orosanu and Connolly (1993). Ill-structured problems, is the first characteristic of a naturalistic decision setting. Other characteristics are uncertain dynamic environments, shifting or competing goals, action/feedback loops and time stress. Thus, NDM better describes the setting in which design decisions are made than does traditional decision making theory. NDM further goes on to describe the decision making process as such, some of which characteristics correspond to the descriptions of the design process provided by Rittel and Löwgren. They are:

- decision makers (particularly experts) frequently generate and evaluate one single option rather than analyse multiple options concurrently
- decision makers choose a solution that is good enough, though not necessarily the best
- reasoning is schema-driven, i.e. guided by the decision maker's knowledge, to search and assess information and to build causal models of events
- deciding and acting are interleaved

Thus, the design process in software development is not a simple matter of defining the problem, outlining the available solution options, comparing them to one another and then selecting the one that results in the best possible usability. It is, indeed, a very complex process, where the outcome depends highly on the skills and insight of the person creating the design. Furthermore, usability is just one out of many aspects that must be weighed into each design decision. As pointed out earlier, interaction design encompasses almost every aspect of a system. Thus, it takes place continuously, at all levels and in all parts/tiers of the system, involving almost everyone in the project. To complicate matters further, many design

decisions are “implicit” and “non-intentional”, in that the decision maker is acting without knowing that he/she is actually making a decision that will affect the usability.

The study

The interviews

The study consisted of a series of interviews with a number of people involved in software development projects in two in-house development organisations (see The organisations and projects). I used two different interview guides, containing a number of topics, and a number of open-ended questions to support the exploration of the topics. The reasons why I chose this approach are:

- This study is exploratory in its nature and qualitative aspects are central. It is not meaningful to collect quantitative data.
- The purpose of the study is to describe how these organisations work with interaction design and usability, and to identify what aspects or factors are important. It is yet too early to test any hypotheses.
- I have certain preconceptions about definitions and the theoretical basis of usability and interaction design. Thus, it is natural to use these definitions and concepts as different topics in the interview guide.

I interviewed eight people in two large government organisations in Sweden. The interviewees had backgrounds in usability, user interface (UI) design and/or software development, see the table below for details.

Interviewee	Usability/HCI	UI design, e.g. prototyping	Software development/programming
I1		X	X
I2	X	X	
I3	X	X	
I4			X
I5		X (web design)	
I6			X
I7			X
I8 – RUP expert			

Each interview lasted about 1 hour. The interviews were tape-recorded, and I took notes on a computer during the interviews.

The interviews were conducted with the aid of an interview guide. The interview guide for the first interview covered the below topics:

- The software development process, in particular in regards to the interaction design and development
- Usability expertise and user involvement
- Examples of design decisions

In the sub-sequent interviews I omitted the last topic. Phrasing the questions in terms of interaction design decisions requires that such decisions are intentional and explicit. Since the first interview indicated that that is often not the case, I decided to modify the interview guide. Instead I included the below topics:

- What input, constraints and support do you have when designing the interaction
- How is the design documented and communicated to others

I phrased the questions slightly differently depending on the background and role of the interviewee. I did, for instance, avoid the term interaction design when talking to non-HCI people.

Weaknesses in the interview approach

Interviewing is prone to a number of weaknesses as discussed by Lantz (1993).

- Interviewing relies on the interviewee's ability to recall a certain situation. This inevitably leads to an interpretation of the information in the interviewee's memory. Thus, what the interviewee says is not necessarily exactly what happened.
- Interviewing requires superb listening skills and skilful personal interaction. The interviewer influencing the interviewee, and thereby the results, cannot be completely avoided.
- The preconceptions of the interviewee may influence the interview so as to bias the results of the interview.

Interviewing is time-consuming, both to carry out and to analyse. Thus, I had to limit the number of interviews, reducing the reliability of the results.

Data analysis

The interviews were tape-recorded and subsequently summarised in writing. Patterns and recurrent concepts were identified and the data was coded in accordance with these patterns. The data was then rearranged to reflect the patterns and concepts, and sorted into structures such that conclusions could be reached. To a large extent, the concepts and patterns emerged during the analysis, although, some of them are based on the questions I had in mind when approaching this matter.

Interaction design and design decisions in these two organisations

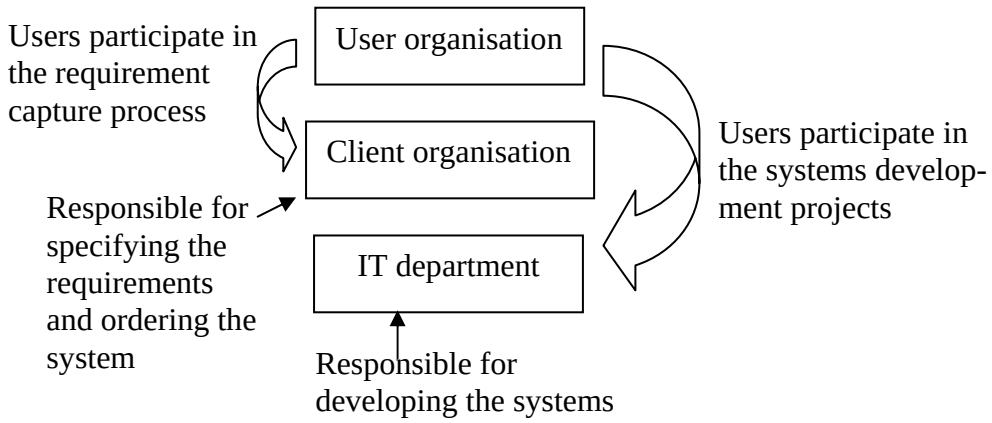
Below, the results of the study are presented, organised in accordance with the concepts and patterns that emerged during the analysis of the data. Thus, a certain amount of processing and restructuring has gone into the presentation. For information and access to the results as raw data, please contact the author.

The results have been grouped in the following categories

- organisations and projects – background information
- software development model – as described from an interaction design point of view
- user-centred design – the compliance of the software development models to the principles of user-centred design
- interaction design – concrete questions and answers about how they work with interaction design, e.g. input, constraints and support
- use cases – use cases turned out to be of great importance for the interaction design and are therefore treated as a separate category
- usability designer – an approach to integrating usability in the software development model

The organisations and projects

Both organisations are large government organisations with in-house IT departments. The IT departments work with bespoke software development, for the parent organisation only. Both organisations have adopted Rational Unified Process (RUP) as their software development model. Both organisations have similar models for the client-developer relation. The below illustration provides a simplified representation of that relation.

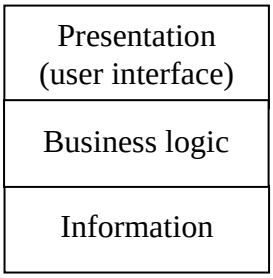


Since the IT departments are in-house, one could assume quite a high degree of familiarity with the context of use. Both IT departments are, however, rather dependent on external consultants with little or no knowledge about the user organisations and context of use.

One of the organisations has a usability team. They attempt to introduce the role of the usability designer, as specified by Göransson and Sandbäck (1999), into the RUP model. The goal is to have one usability designer working part time or full time in all the large projects. This role will assume the responsibility for the interaction design. In small projects, usability work is carried out as single activities, for instance, user analysis or usability evaluations. The other organisation does not have a usability team, but are investigating the possibilities of introducing one. There, the user interaction is usually designed and developed by software

engineers, with a special interest and skills in user interface design and usability. Little time is spent on traditional usability activities.

Both organisations use the three-tier system architecture. The three-tier architecture divides the system as illustrated below:



The presentation tier (user interface) in the architecture utilises services provided by the business logic tier. This means that the presentation tier communicates and interacts with the services in the business logic tier and the information tier to provide the features, functions, dialogues and information with which the user interacts to perform his/her tasks and accomplish his/her goals.

Software development model

Both organisations use Rational Unified Process (RUP) to control their software development. It is not within the scope of this paper to describe RUP but it is important that the reader understands the concepts of use cases. Use cases are an essential part of RUP, they describe the scope, contents and to a certain extent the behaviour of the system. Use cases are developed in modelling sessions, often together with the users, and are a way of capturing the user requirements. For more information about RUP, please refer to for instance Kruchten (1998).

Below are described certain aspects of the software development process that may be of interest from the interaction design point of view. These matters are particular to these organisations, and should not be seen as representative for RUP in general.

In large projects, work is normally organised in accordance with the three tiers. One team (user interface developers) builds the presentation tier, based on services provided by another team, the business logic team. Yet another team develops the information tier. In small projects, the same person may develop all three tiers, or two of them, for a particular function. In addition to this division of work, different teams usually develop different parts or packages of the system. The usability team have little to do with the services created by the business logic team. Their efforts focus on the user interface design and development.

The basis for the software development project is some kind of requirement specification or description from the client organisation. Users and/or domain experts[1] participate in the requirements capture process. The development project usually starts with use case modelling, resulting in a use case model. The use case specifier and client representatives,

[1] Domain experts are representatives from the user organisations with expertise in the domain, e.g. laws and regulations affecting the new system. These people will not necessarily be active users of the system.

users and domain experts, usually perform this modelling. The user interface developers often participate in the modelling sessions, but not always.

The user interface modelling is done in parallel with the use case modelling, with the same group of users and domain experts. The user interface developers/use case specifiers develop a solution, which is then reviewed by the users and domain experts, and modified accordingly. In one of the organisations, the client organisation sometimes does the user interface modelling.

The design solutions are often iterated or reviewed a few times, before they are implemented. The number of iterations/reviews is determined by the date the implementation must start. Proper usability tests are run in the organisation with the usability team.

User-centred design

In order to understand the interaction design process in these organisations, I asked a number questions concerning the software development process, particularly in regards to interaction design, user participation, iterations, etc. Below, these results are presented, structured in accordance with the four principles for user-centred design or design for usability as defined by Gould, et al.

Early focus on users

The first of these principles is early and continual focus on the users.

Both organisations have user and domain expert participation in the development; primarily in the requirements capture process, the modelling and the analysis and design activities. Both organisations base their user participation mainly on permanent user representatives, i.e. people in the user organisations that have been promoted to a “user representative position” in the client organisation. The projects also work with end user representatives directly from the user organisations. Usually, the same users participate throughout the development, and they may be removed from their work for a couple of years in a long project.

Quite a few of the interviewees reported a certain amount of frustration with the user participation. One recurring complaint was that some user representatives are unable to envision new ways of doing things. User representatives that want change and that are open-minded about new technical possibilities were considered fun and stimulating to work with.

One interviewee complained that the user representatives often become a bottleneck in the development process. There are not enough of them to “go round” all the teams that need information from them.

One interviewee felt that too many users in the modelling sessions makes communication cumbersome and time-consuming.

Yet another grievance was that the user representatives sometimes veto modifications and suggestions. Even when a modification is suggested for usability reasons by the designers or developers. Some interviewees felt that the best way to deal with this problem, would be to create a design solution and then show it to the user representatives or possibly run an evaluation.

Empirical development

The second principle is early and continuous user testing, that is, all design solutions should be evaluated regarding usability.

Evaluating your design solutions requires prototypes, lo-fi or hi-fi. Prototyping is used in the projects discussed here, at least in a very liberal sense; such as drawing design suggestions on the whiteboard together with the users or having the user representatives reviewing dialogue specifications. The prototypes are used to try out design solutions and check them with the user representatives, as well as checking the correctness of the use case model. The prototypes are made on paper as well as on computer.

Few of the projects in the interview study run proper usability evaluations on the prototypes. One of the interviewees said that they normally run simple evaluations with the user towards the end of the analysis phase to check the workflow. The evaluations are based on the use cases. Naturally, projects where the usability team is involved use evaluation techniques to a higher degree.

The interviewees described different reactions to the prototypes. When the user needs are obscure, the prototype becomes a tool for generating requirements, involving major changes to the first suggestions. When the requirements have been fairly well specified in advance, changes to the prototype are on a more detailed level.

None of the organisations work with explicit, quantified, usability requirements.

Iterative development

The third principle is iterative design where the key requirements are that the required changes must be identfied, and that the project has the ability and is willing to make those changes.

In both organisations, the interviewees described their work being divided into packages or releases, i.e. parts of the system that are to be delivered to the users. None of them expressed their work process in terms of iterations. On the contrary, some said that they did not quite understand what the iterations are about. They did not worry too much about that though, they were happy with working with packages.

According to one of the interviewees, the iterations cover several packages, and the primary aim with the iterations is to make sure that all the teams have “something to do”. Iterations were described by this interviewee as a tool for the project management to manage the resources within the project. The design loops - i.e. design suggestions, evaluations and modifications – are not considered iterations in the RUP meaning of the term. Design loops stop when the date for implementation is due

Integrated design

The fourth and last principle is integrated design, i.e. that all aspects of usability evolve in parallel. Integrated design requires that one group of people be given full responsibility for all usability aspects.

There is no role for the overall responsibility of the interaction design. Instead that responsibility is divided between the teams, the user interface team doing their part, and the business logic team doing theirs. Interviewees from both organisations talked about the user

interface team and the business logic team working in isolation from one another. They also mentioned the users or the client representatives bridging the gap between the two. The usability designer is introduced to assume the responsibility for the interaction design and usability work.

The separation creates un-necessary misunderstandings and problems, such as, the use case modelling people being unable to describe the user needs in such a way that the developer can design and implement the user interface or business logic. In addition to that gap, one of the organisations has divided the user interface development further into design and implementation, where the use case people do the design, and the user interface developers do the implementation. The means of communication is a detailed description of each dialogue in the system.

The interaction design

The main part of the interviews was spent on discussing the interaction design, how do they work with design, who makes the decisions, what controls the decisions, what support do they have when designing the interaction, etc? The results are presented below.

Input for design

The main input to the interaction design is the use case model. Requirements are captured through use case modelling and expressed mainly by use case models. The use case models are later refined to analysis models and design models. However, the use case model does not provide much support for working with the usability of the system. RUP has specified other techniques to be used in conjunction with the use case model to cater for the usability aspects, for instance, storyboards. These organisations, however, do not use these “extra” techniques. Use case models are discussed in detail further down.

In addition to the use case model, the developers have access to users and domain experts.

The usability team have tried another approach in a “usability designer pilot project”. The usability designer works with participatory prototyping in parallel with the use case modelling. The input to the design workshops is personas and a context of use analysis. A prototype is created together with users in workshops. This prototype will be used as a model for the system implementation and integrated with the user interface style guide.

Dialogue specifications are used in one of the organisations. They are, however, not input to the design, but output. The user interface developers use the dialogue specifications as input to the implementation of the user interface.

The interviewees also have an existing system where the new user interface is to be incorporated. Nevertheless, one interviewee complained about not knowing what to do about, for instance, navigation between dialogues or between tabs in a tabbed window. Neither the use case models, nor the dialogue specifications cover that. The projects have to make up their own guidelines regarding such matters.

In one project, they started out with designing the main window, from which the user starts all other actions. The design of the main window was determined by the user representatives. They had a very clear picture of what metaphor they wanted and how they wanted to use it.

Who decides in design matters?

The interviewees were asked what kind of decisions they are allowed to make, and whether design decisions are made by them or by other people.

The type of communication with the client seems to influence how interaction design decisions are made. In one case, communication with the clients was very informal, there were, for instance no formal requirements document. In this case, the designers and developers had a high degree of freedom to make the interaction design decisions.

In other projects, where communication is more formalised, by means of, for instance, requirements specifications and use case models, the interviewees reported less freedom to decide on the interaction. The client organisations tend to describe the systems at a detailed level in their requirement specifications or use case models. In some cases, the requirements talk more about what the system should look like, than what the business looks like.
–*"The requirement specifiers want to work with software development"*.

In some cases, the client organisations specify the user dialogues (windows), leaving the developers little freedom regarding the interaction design. The developers are able to argue for modifications by presenting other design suggestions or by referring to the style guide. One consequence of this work division is that the client organisation regards the documents as "theirs" even when they have handed over the responsibility of the documents to the development team.

The responsibilities of each party (client organisation and development organisation) are sometimes a bit unclear. Some of the interviewees felt that the client organisations get involved in areas and with responsibilities that "belong" to the development organisation, for instance, writing dialogue descriptions or making interaction design decisions.

Another aspect that affects the freedom to make design decisions is whether the design and implementation is done by the same person, or by different people. In one of the organisations, the user interface is designed by the use case specifier together with the user representatives (or by the client organisation). The developer implements the design as described in a dialogue specification.
In other projects, the same person does the design and the implementation. Issues regarding the design can be resolved directly with the users in the user interface modelling sessions.

When asked who makes the design decisions, none of the interviewees said "I do". Instead, they brought up the user representatives and the client organisation. Nearly all the interviewees pointed to the user representatives and/or client representatives in the projects.
– "*We (the development team) can't do anything they (the user representatives) don't agree to.*"

According to several of the interviewees the usability of the resulting system depends to a large extent on the user representatives, for instance, to what extent they are able to let go of the old routines and ways of interacting with computers. One of the interviewees claimed that user representatives often want to build in too much control into the system, i.e. allowing the users little control over the tasks and workflows.

The user representatives sometimes veto modifications and suggestions, even if the designers suggest them for usability reasons. The interviewees reported trying to get around this problem by not asking the user representatives about details in the design any longer.

Naturally, the user interface designers/developers do make design decisions. Quite a few of the design decisions are made in workshops (modelling sessions) together with the user representatives. Some of the interviewees reported difficulties with keeping the modelling sessions at an appropriate level. One problem was too many participants. Another problem was getting into drawing details of the user interface too soon.
–"To start with, we used to draw on the whiteboard, but there were too many detail matters, such as OK on the button. There were too many people (appr 10 participants), we got nowhere."
They changed that. Normally, there are no more than 5-6 participants, and they (the user interface designers) are clear about what kind of information they need. The interface designers, with the support of the usability people, make the design decisions.

Finally, there are matters over which the development projects do not have control, despite the fact that these matters are likely to affect the perceived usability of the system. One such matter is a general case handling routine being deployed in one of the organisations. The routine will be supported by a bespoke system. Not all units within the user organisation comply with the general routine. Thus, the system will more or less force those units into complying with the routine.

Business logic versus user interface design

Since the business logic tier provides the services, which are utilised by the presentation tier (user interface), one of the concerns in this study was to find out to what extent the interaction design is determined by the business logic tier.

The use cases are the basis of the design for all teams. The teams work in parallel and develop separate analysis models and design models. The services and information provided to the user are designed to meet the requirements on the presentation.

The user interface team and the business logic teams do the modelling together with the users but in separate sessions. The user representatives act as go-betweens to a certain extent in between modelling sessions. Cooperation between the user interface team and the business logic team is a two-way affair. The user interface prototype sets the requirements on the business logic services and they are designed and implemented in accordance with these requirements. Then the final user interface is implemented against these services.

Support

What support regarding usability do the designers/developers get when designing the interaction?

In both organisations, there are people who are seen by the others as someone "who knows something about usability or design". They attract questions from the developers about design matters.
– "I am one of those people that the developers call to check things that have to do with design."

In the organisation with the usability team, the user interface designers/developers turn to them in matters of design. Even if they do have a style guide in the organisation and project specific guidelines, most of the support comes from the usability people.
–*"No two cases are alike. It is difficult to see yourself whether it is usable or not. We sit down with the usability people and discuss. Support in usability matters comes primarily from the usability team."*

The other source of support is style guides and design recommendations. The user interface designers and developers often refer to the style guide and the project specific recommendations. They avoid discussing such matters with the user representatives. Style guides are seen as a means for preserving consistency over different applications. Regarding the quality of the recommendations opinions varied.

Another interviewee said that the style guide is based on a particular way of creating the dialogues, which this person found inadequate, even though the actual recommendations regarding the design are good. Another problem with the style guides is that they necessarily reflect a particular target environment.

Yet another interviewee appreciated the style guide. It contains the right kind of information, about for instance, what should happen when the user clicks the Cancel button, which menus are standard, and which are extra menus, what access keys to use, etc. Whereas it leaves matters such as colours fairly open.

Constraints

The interaction design is partly determined by the conceptual model, style guides, style matters imposed by the client (regarding for instance colours and logos) and main user interfaces, as well as the target environment style. Naturally, usability issues are always traded off against, for instance, performance or other technical matters.

In addition to those aspects, the interviewees reported both "external" and "internal" constraints in the design situation.

One recurring complaint was that user representatives are sometimes unable to envision new ways of doing things. The interviewees also talked about getting "blocked" by old and well-used solutions to similar problems.

One interviewee pointed out that personal attitudes determine to what extent the individual developer tries to comply with the users' wishes. Some developers are more accommodating towards users and their problems, whereas others tend to see user involvement as disruptive.

The development environment, naturally limits the designer in the design. It did not seem a major problem, though. When it happens, the interviewees seemed to accept it with an "it's all in the day's work for the software developer" attitude. One interviewee, notably one of the usability team members, did talk about the impact the choice of development environment has on interaction design and usability. Also the target environment limits the design space.

Use cases

Requirements are captured through use case modelling and expressed mainly by use case models. Thus, the use case model is central to the whole software development process. But,

use case modelling is by no means a trivial way of capturing requirements. Several difficulties and shortcomings were reported.

The interviewees usually work with a set of use cases, picked out by the project management and the client organisation, for a particular release. When asked if they had seen the whole use case model to get an overall picture of the system, most of them said they had not.

The interviewees in both organisations reported difficulties with the use case modelling. One of the interviewees described a project where the use cases were rewritten several times. The first lot of use cases were not usable (written by the client organisation) – they did not contain the required information. The second lot (written by the client organisation) were far too detailed; the third lot (written by the development team) has been used for the design and the development of parts of the system. They are now to be rewritten once more.

Another interviewee reports difficulties with understanding what a use case is. That developer claimed that the project members have misunderstood the concept, resulting in far too many, far too detailed system operations instead of real use cases.
"I still don't know what a use case is."

Despite the difficulties with producing the use cases, they provide at least some of the information that is required for the design of the interaction. That is, the use cases specify what the user should be able to do in the system and also how.
"*A use case should really capture what and how the users really work."*

Some of the interviewees appreciated that the use cases provide a step-by-step description of the user interaction with the system. They seemed to equate this description with the workflow of a real user task. Regardless of how truthful the description is to the real task, it does provide information about the use, which has been missing in earlier models.

Some interviewees reported problems with the use cases being far too detailed about the user-computer interaction (in terms of clicking buttons, etc). To avoid this, they specify the user-computer dialogue in a separate document

In other projects the use case model is used as a basis for the user interface design. The user interface is modelled straight from the use cases, each use case corresponding to a particular set of windows or dialogues. The use case model and the user interface design are developed in parallel, iteratively, where the one affects the other and vice versa. They do the user interface design use case-wise.

In both organisations, use case modelling is done in sessions with the domain experts and user representatives. In some cases, the client organisations write the use cases, which are then elaborated together with the software development project. No matter in what form the original requirements come, the use case model is the major means of communication with the user representatives.

Summary and conclusions

The research questions were:

- Who decides on matters that are connected to usability, such as, what services will be provided by the system, how these services are represented, and what information is available to the users?
- What support regarding usability matters does the decision maker have access to?
- What is the major input to these decisions?
- What are the constraints?

Since the interaction design process is a "fuzzy" process dealing with poorly defined wicked problems, answering these questions is no simple matter. The results do point in certain directions, as discussed below.

There is no straightforward answer to when the decisions affecting usability are being made. These decisions encompass all levels from what is to be included in a certain release (i.e. what the user will be allowed to do in the system, in terms of tasks or functionality) to the use case modelling, to detail decisions about input/output controls. User interface modelling seems to be considered part of the requirements capture process or use case modelling, and is carried out fairly early in the projects. But, there are changes throughout the projects. Thus, there are changes in the design that affect the usability throughout the project.

In the organisations in the study, interaction design is often split up and done tier-wise in accordance with the three-tier system architecture. The major means of communication is the use case model, and modelling sessions with the user representatives. Thus, there is nobody who assumes the overall responsibility for the interaction design. RUP does not define such a role or such a responsibility. The usability designer may be such a role but it is too early to tell yet.

The interviewees claim that the user representatives and domain experts in the projects are responsible for most of the interaction design decisions. Naturally, a lot of the decisions are made by the designers/developers. However, the interviewees did not answer, "I do" when asked about who makes the decisions. This phenomenon may be explained by the fact that in many cases design decisions just "happen" as part of a process, where the design slowly grows and develops. Design decisions are explicitly made in those cases where a design issue has been discussed, for instance, when not complying with the style guide or when there has been disagreements. It is interesting to note, in connection with this, that Löwgren, et al, define design as a political and ideological activity in that the resulting artefact affects the freedom of action of other people. They go on to claim that the designer is *always* responsible for the resulting design. It seems that the interviewees do not agree with this view of design. And in cases, where the client/user representatives actively intervene in design decisions, as described by some of the interviewees, placing the responsibility on the designers/developers alone does not seem fair.

Another matter that may explain the fact that the interviewees considered the user representatives making most of the design decisions may be that quite a few of the decisions are implicitly made in modelling sessions together with the user representatives and domain experts. Moreover, the interviewees based their designs on the use case model, which is developed together with the user representatives.

It seems that the interviewees are more open to suggestions regarding design matters from the user representatives, than if the user representatives actually make the decisions. This may be a matter of "not invented here" or the interviewees feeling that they are the ones with the expertise in interaction design, not the users.

The support regarding usability matters is primarily provided by style guides, user interface recommendations and by the usability team. In the organisation with the usability team, they were considered more helpful than the style guide. Both the style guides and the usability team provide support for the user interface design only.

The constraints in a design situation are both internal – being stuck with certain design ideas – and external – such as style guides, main user interfaces and development environment. The internal limitations seemed to be more frustrating to the interviewees, than the external ones.

The interview results indicate that the business logic development places little or no constraints on the user interface design. It is rather the user interface design that controls the design of the business logic, in making requests for information being sent from or to the business logic components.

The use cases are the most important information carrier in the projects. They are the basis of the interaction design and thus limit the design "space". They specify, for instance, the workflow, what information should be displayed when, how the information should be grouped, etc.

The, perhaps, most crucial constraint, however, are deadlines. When the delivery date comes close, quality aspects and usability aspects go out the window. Interaction design is reduced to getting things into place, rather than trying to find good design solutions promoting usability.

When looking at the design decisions made in these organisations – some problems may be referred to problems with applying a user-centred design approach, as defined by Gould and Lewis. They were

- Early focus on the users
  These organisations have chosen to work with more or less permanent user representatives, which are removed from their ordinary work for long periods of time. This approach seems to create problems, not only because the users are not particularly representative after a period of time. The interviewees seemed a bit frustrated with the way a few users/domain experts influence the design solutions to a very high extent. They were also irritated with some of the user representatives being unwilling to let go of old routines and ways of working with computers.
- Iterations
  Since design problems often are "wicked problems", solving them requires an iterative approach. But, it seems that applying the iterative approach has been and is difficult for these organisations. Does that mean that the iterative approach as such is not applicable in this particular kind of development, or does it mean that RUP applies the concept in a confusing manner?
- Empirical design
  Only one of the organisations run usability evaluations with users. None of the organisations measure usability.

- Integrated design
  Bespoke software must be developed in parallel with e.g. routines, roles and skills in the user organisation. The software development project seldom has impact on those matters. Thus, matters that are essential to the perceived usability of the system are often not controlled by the software development team.

Conclusions

What would keep usability in focus in projects such as the ones discussed in this study? What would help the designer or developer make design decisions so that the resulting system becomes usable to its users? The interviewees themselves pointed out certain measures they felt might help matters. They were:

- Having rather small, well-balanced groups in modelling sessions, as regards the number of users versus the number of people from the development team. The users must be open-minded and willing to change routines and work roles.
- Having software developers in the modelling sessions, in addition to modelling people. The software developers bring in knowledge about technical possibilities and limitations.
- Include the software developers in evaluation sessions.

Communication is one of the underlying bases of these matters. Communication was pointed out as crucial for success, including informal communication. It is important that the projects are organised so as to facilitate that communication, by, for instance, placing the whole project together in one open area.

It is not clear who is responsible for the interaction design, which might in the end result in nobody assuming that responsibility. The matter may be improved by the introduction of a usability role in the projects (usability designer). The responsibilities and activities of this role should include the overall responsibility for the interaction design and usability, as well as the responsibility for the user-centred approach in the project. This automatically means that other roles, including developers and users, are not allowed to ignore or undo the decisions of the usability designer. Such a role should probably be part of the project management.

Use case models are crucial to the projects described in this study. They are the prime information carrier and the basis of the interaction design. A poor use case model, will inevitably lead to a system with poor usability. In fact, use-case driven design does not in any way guarantee good usability. Lif (1998) describes how use case models often result in a fragmented user interface where each use case is represented by its own set of dialogues. Usability matters should therefore be made a part of the use case modelling by, for instance, involving the usability people in the modelling sessions.

Another aspect of use cases is the difficulties in getting them right. If they are to be the major information carrier in a project, more time should be spent on understanding how to write them. It is important that they provide the information required to create a usable design. If use cases are difficult to get “right”, maybe they are not the best means of capturing user requirements.

Another important matter is that users should participate throughout the project, and not just in modelling sessions and acceptance tests. The development team need to be able to talk to users on short notice, since questions about the context of use keep coming up throughout the design and implementation.

Van Welie, et al, point out that knowledge about the context of use is essential. In these organisations, different people often perform the analysis, the design and the implementation. Information is bound to get lost in that process. Therefore, it is important that the people designing and implementing the system have direct contact with the users.

The usability team was considered a better support in usability matters than the style guide. The style guide can never provide complete answers to all the questions the designer or developer faces when creating the interaction. Therefore, the usability team members should spend as much time as possible with the different projects - "Usability by walking around".

Finally, it is important that the iterative approach is made clear to the development projects, so that they can run proper iterations with design suggestions, usability evaluations and modifications, in close co-operation with the users. This is an important aspect, in particular since the definition of iterative development in RUP (see below) does not seem to comply with the one provided by Gould, et al.

Iterative development in RUP as defined by Jacobson, Booch and Rumbauch (1999)
"In the context of the software life cycle, a process that involves managing a stream of executable releases." (Appendix C General Glossary, p 446.)
and iteration:
"*A distinct set of activities conducted according to a devoted (iteration) plan and evaluation criteria that results in a release, either internal or external." (Appendix C General Glossary, p 446.)*

Applicability and validity of the study

Trying to identify how usability is addressed by investigating who makes the interaction design decisions is a formidable task, since design affecting usability takes place continuously at all levels in the software development process. Design, is not a well-defined, structured process. Moreover, it is most likely particular to each project, formed by the software development model used, and the people involved in the project. Thus, the results reflect the organisations and software development models used in the projects in this study. Therefore, the results cannot be generalised, but they may, nevertheless, contribute to the general understanding of how usability is taken into account in interaction design.

One of the weaknesses of the study is that, facing the question "Who makes the design decisions?" few people would say, "I do" – even if they do make design decisions. Many design decisions are likely to be non-intentional, i.e. the person making the decision is not aware of making that decision, much less that the decision has anything to do with usability. Identifying and investigating such non-decisions about design is difficult, requiring far more time and effort than allowed for in such a limited study as this. Thus, the results of this study must be seen as tentative, pointing out areas and issues to be further looked into.

Future work

Since this study was exploratory, the above results must be corroborated in future studies, for instance, in

- Interviews with designers and developers in other types of organisations
- Observations of modelling sessions and other workshops or sessions with designers/developers and user representatives

The results do point to certain matters that would be of interest to investigate further, for instance:

- User representation and participation – on what terms should the users participate in the design process, so that their contribution becomes constructive and well balanced?
- What support do the designers/developers need when making interaction design decisions?
- Usability people usually concentrate on the user interface design – working closely together with user interface teams in software development projects. But interaction design encompasses all parts of the system – should the usability people be involved in the business logic design, and database design?
- Usability activities are typically concentrated to the early phases of a project, such as, requirements capture and prototyping. But, changes that affect the usability happen throughout the project. How should usability work be integrated with all the phases of the project?

Acknowledgements

This work was performed with financial support from the Swedish council for Work Life Research in cooperation with the Swedish National Tax Board (Riksskatteverket – RSV) and the Swedish National Board for Social Securities (Riksförsäkringsverket – RFV). All the time spent and expertise shared by the interviewees (software developers and usability people) from these organisations is greatly appreciated.

References

- Allen, P. Frost, S. (1998). *Component-Based Development for Enterprise Systems – Applying the Select Perspective,* Cambridge University Press, New York, USA
- Clegg, C. Axtell, C. Damodaran, L. Farbey, B. Hull, R. Lloyd-Jones, R. Nicholls, J. Sell, R. Tomlinson, C. (1997). Information technology: a study of performance and the role of human and organizational factors. *Ergonomics*, Vol 40. No 9. 851-871
- Dix, A. Finlay, J. Abowd, G. Beale, R. (1998). *Human-Computer Interaction*. Prentice Hall Europe, Bath, the UK
- Göransson, B. Sandbäck, T. (1999). Usability Designers Improve the User-Centred Design Process. Brewser, S. Cawsey, A. Cockton, G. *Human-Computer Interaction – INTERACT '99 (Volume II),* Edinburgh Press, the UK.
- Gould, J.D. Boies, S.J. Ukelson, J. (1997). How to Design Usable Systems, in Helander, M. Landauer, T.K. Prabhu, P. *Handbook of Human-Computer Interaction*, Second revision, Elsevier Science B.V., Amsterdam, the Netherlands.
- Gulliksen, J. Göransson, B. (2000). *Användarcentrerad systemutveckling*. Technical Report 2000-004, Uppsala University, Uppsala, Sweden.
- International Standards Organization. (1998). *ISO 9241: Ergonomic requirements for office work with visual display terminals.* Geneva, Switzerland.
- Jacobson, I. Booch, G. Rumbauch, J (1999). *The Unified Software Development Process,* Addison Wesley Longman, Inc, Reading, Massachusetts, the US.
- Kruchten, P. (1998). *The Rational Unified Process – an Introduction*, Addison Wesley Longman, Inc, Reading, Massachusetts, the US.
- Lantz, A. (1993). *Intervjumetodik*. Studentlitteratur, Lund, Sweden.
- Lif, M. (1998). *Adding Usability - Methods for Modelling, User Interface Design and Evaluation*. Eklundshofs Grafiska, Uppsala, Sweden
- Löwgren, J. Stolterman, E. (1998). *Design av informationsteknik – materialet utan egenskaper,* Studentlitteratur, Lund, Sweden
- Orosanu, J. Connolly, T. (1993). The Reinvention of Decision Making. In Klein et al (Red.). *Decision making in action: models and methods*. Ablex Publishing Corporation. Norwood, NJ, the US
- Rittel, H.W.J. Webber, M.M. (1973). Dilemmas in a General Theory of Planning. *Policy Sciences*, 4, 155-169. Elsevier Scientific Publishing Company, Amsterdam, the Netherlands.
- van Welie, M. van der Veer, G.C. Eliëns, A. (1999). Breaking Down Usability. In Sasse, M.A. Johnson, C. *Human-Computer Interaction – INTERACT '99*. IOS Press, the Netherlands.

Technical reports from the Department of Information Technology

2001-021 **Inger Boivie:** ***Usability and Design Decisions in Software Development***

2001-020 **Emmanuel Beffara and Sergei Vorobyov:** ***Adapting Gurvich-Karzanov-Khachiyan's Algorithm for Parity Games: Implementation and Experimentation*****.**

2001-019 **Wendy Kress and Jonas Nilsson:** ***Boundary conditions and estimates for the linearized Navier-Stokes equations on staggered grids***

2001-018 **Emad Abd-Elrady:** ***An adaptive grid point RPEM algorithm for harmonic signal modeling***

2001-017 **Henrik Björklund, Viktor Petersson and Sergei Vorobyov:** ***Experiments with Iterative Improvement Algorithms on Completely Unimodal Hypercubes***

2001-016 **Robert Stjernström:** ***User-Centred Design of a Train Driver Display***

2001-015 **Magnus Svärd:** ***On coordinate transformations for summation-by-parts operators***

2001-014 **Paul Pettersson and Sergio Yovine:** ***Workshop on Real-Time Tools (Proceedings)***

2001-013 **Eva Olsson, Lena Kecklund, Michael Ingre and Anders Jansson:** ***Lokförarens informationsmiljö och ATC. Ett användarperspektiv***

2001-012 **Friedhelm Stappert, Andreas Ermedahl and Jakob Engblom:** ***Efficient Longest Executable Path Search for Programs with Complex Flows and Pipeline Effects***

2001-011 **Torsten Söderström, Umberto Soverini and Kaushik Mahata:** ***Perspectives on errors-in-variables estimation for dynamic systems***

2001-010 **Thiemo Voigt and Per Gunningberg:** ***Dealing with Memory-Intensive Web Requests***

Oktober 2001
ISSN 1404-3203
http://www.it.uu.se/