WORKSHOP 13

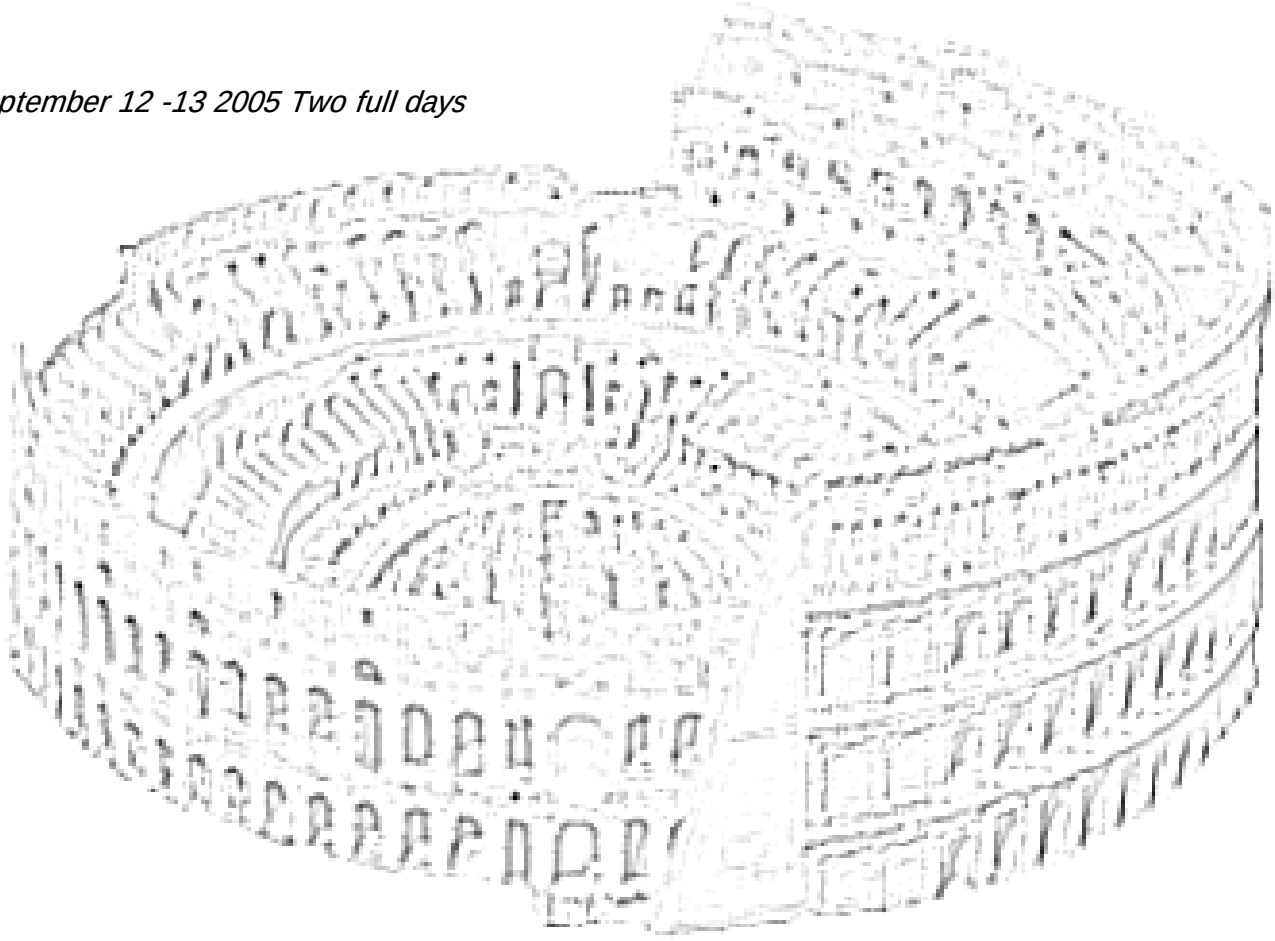
INTERNATIONAL COST294 WORKSHOP ON USER INTERFACE QUALITY MODELS

COST – European Cooperation in the field of Scientific and Technical Research

COST294 - MAUSE

Jean Vanderdonckt, Effie Lai-Chong Law & Ebba Thora Hvannberg

September 12 -13 2005 Two full days



In conjunction with:

Tenth IFIP TC13 International Conference on Human-Computer Interaction
12-16 September 2005, Rome, Italy

INTERACT 2005

ACKNOWLEDGEMENT

First of all, we are grateful to the organizers of INTERACT 2005, who have granted us the valuable opportunity to hold our Workshop "International COST294 Workshop on User Interface Quality Models (UIQM)" in conjunction with the conference, and also provided the timely support for printing this Workshop's Proceedings. Thanks must also go to the authors of the Workshop papers, whose contributions serve as rich sources of stimulation and inspiration to explore the issues of interest from multiple perspectives. The quality of the contributions could further be ensured and improved with the generous help of a group of reviewers, who are leading researchers and experts in the field of usability evaluation. Their prompt responses to our calls for peer review, and their effective as well as efficient review works are highly appreciated.

List of the reviewers of Int'l COST294 Workshop on UQIM (2005)

Name	Affiliation	Country
Abrahão, Silvia	Technical University of Valencia	Spain
Bernhaupt, Regina	ICT&S/ Universität Salzburg	Austria
Blandford, Ann	UCLIC	UK
Carr, David	Lulea University of Technology	Sweden
Cockton, Gilbert	University of Sunderland	UK
Følstad, Asbjørn	SINTEF	Norway
Guttormsen, Sissel	Universität Bern	Switzerland
Hornbæk, Kasper	University of Copenhagen	Denmark
Hvannberg, Ebba T.	University of Iceland	Iceland
Jokela, Timo	University of Oulu	Finland
Lárusdóttir, Marta	Reykjavik University	Iceland
Law, Effie L-C.	ETH Zürich	Switzerland
Norros, Leena	Technical Research Centre of Finland	Finland
Palanque, Philippe	LIIHS-IRIT Université Paul Sabatier	France
Paternò, Fabio	HIIS Laboratory ISTI - C.N.R.	Italy
Pribeanu, Costin	ICI Bucharest	Romania
Sikorski, Marcin	Gdansk University of Technology	Poland
Springett, Mark	Middlesex University	UK
Stary, Christian	University of Linz	Austria
Tzanavari, Aimilia	University of Cyprus	Cyprus
Vanderdonckt, Jean	Catholic University of Louvain	Belgium
Zimmerman, Silvia	Institut für Software-Ergonomie und Usability	Switzerland

Last but not least, we express gratitude to our sponsor – COST (European Cooperation in the field of Scientific and Technical Research; http://cost.cordis.lu/src/home.cfm). The COST Office operated by the European Science Foundation (ESF) provides scientific, financial and administrative support to COST Actions. Specifically, the COST Action 294 (http://www.cost294.org), which is also known as MAUSE, was officially launched in January 2005. The ultimate goal of COST294-MAUSE is to bring more science to bear on Usability Evaluation Methods (UEM) development, evaluation, and comparison, aiming for results that can be transferred to industry and educators, thus leading to increased competitiveness of European industry and benefit to the public. The current Workshop is the first open workshop implemented under the auspices of COST294-MAUSE. As with other past and forthcoming events of COST294-MAUSE, we aim to provide the participants with enlightening environments to further deepen and broaden their expertise and experiences in the area of usability.

FOREWORD

Aim of the Workshop

This Workshop aims to bring together researchers and practitioners who have interest and experience in assessing the quality of a user interface of an interactive system. It is also targeted at people designing, developing, using, or testing software that ensures some form of quality for user interfaces, manually, automatically or in a computer-aided way, e.g., logging tools, validity checker, reverse engineering tools, re-engineering tools, transcoders, usability checkers, accessibility verifier, automatic metric capture, statistical tools, and guideline reviewers. The purpose of this Workshop is to establish a basis for a model, qualitative or quantitative, that would help to assess qualities of a user interface, such as accessibility, usability, reliability, and reusability. A framework will help participants to locate approaches and software along several dimensions, to both classify existing related work, and also to discover new dimensions. There will be a specific focus on tools and methods hat helps assess the quality of web sites. Discussion will be structured around an analysis grid for assessing the validity of existing quality models. All attributes relevant to the quality of user interfaces are appropriate, such as those defined in ISO 9126: functionality, reliability, usability, efficiency, maintainability, and portability.

Motivation for the Workshop

Despite abundant usability and accessibility knowledge, the quality of user interfaces continues to be a pressing issue for human-computer interaction. The majority of user interfaces have usability and/or accessibility problems. To achieve software quality for an interactive system, the software's attributes must be clearly defined. Otherwise, assessment of quality is left to the intuition or the responsibility of lead developers. In this sense, a quality model must be built at the outset, and evaluation methods should be used during design and implementation stages and be based on these quality models. Several existing quality models only emphasize some aspects:

- Software Engineering pays most attention to the software quality in terms of factors such as correctness, robustness, usability, extendibility, and reusability, and typical quality factors found in the ISO 9126 standard.
- Software Performance tends to favour performance factors such as rapidity, compactness, and efficiency.
- Assistive technologies consider that the most important quality factor of a web site resides in its accessibility for the widest possible audience (perhaps with a range of disabilities or specific needs), in different contexts of use or on various computing platforms.
- Mobile Computing is interested in factors affecting the code and the contents of a web site to transfer it to a mobile phone, a Personal Digital Assistant (PDA), a handbag PC, a tablet PC or a laptop.
- Graphic Design skills and requirements address the quality of the visual design of web sites.
- Web Engineering applies systematic, disciplined and quantifiable approaches to development, operation, and maintenance of Web-based applications. Developers are concerned with the well-formedness of code with respect to established standards, as well as the validity of the code is also an integral part of the quality.
- Human-Computer Interaction has tackled how to assess the usability and the accessibility of a web site according to user-centred methods.

A key goal of this workshop is to bring together people who have experience in one or several areas. Since there is no universal method for measuring the quality of a user interface in all circumstances, many existing methods approximate quality by different models, involving different types of functions. As quality requires interpretation, the Workshop will address the following goals:

- We must identify the underlying quality models used for evaluation of user interfaces for specific potential quality attributes, e.g. for usability and accessibility. For example, the ISO 9126 standard decomposes quality into six factors that are in turn decomposed into sub-factors; the standard does not precise how these sub-factors can be effectively and efficiently measured. For another example, the standard ISO 14598 Software Engineering – Product evaluation – is written to guide evaluation of software products according to the ISO 9126 quality model, but it is not known how effective and efficient these measures are.

- We need to identify the potential shortcomings of Usability Evaluation Methods (UEMs) associated with these models. For example, designers, developers and evaluators need to understand general characteristics of quality evaluation tools, as they need to know that there is more for ensuring the quality of a website than merely checking it with a tool.
- We need to identify the strengths and weaknesses of tools for automated evaluation. For example, many accessibility tools are based on accessibility guidelines, but vary slightly in their interpretation of guidelines. Hence, these results still require human judgment.

Contributions of the Workshop

Altogether there are 15 contributions authored by 25 researchers coming from different European countries, including: Austria (3), Belgium (3), Denmark (1), Finland (1), Iceland (2), Italy (3), Norway (1), Poland (1), Romania (2), Spain (6), Sweden (1), and Switzerland (1). The contributions are relevant to the overarching theme of the workshop. Subsets of contributions address a closely related topic, but tackle it from different perspectives and with different methodological approaches. Consequently, the 15 contributions are divided into four theme groups, on which the order of presentation and subsequent discussions are based, thereby facilitating the participants to come grips with multi-perspective arguments of a specific theme and sharpening the focus of discussion. Specifically, the four theme groups are:

- **Group 1: Usability Evaluation** – Several intriguing topics related to the methodological approaches of usability evaluation are addressed in the contributions of this theme group, including the effectiveness of usability inspection methods (UIM), the analysis of return on investment (ROI), the tool and method for remote usability evaluation (RUE) and Statechart Metrics for usability evaluation (ScM). In the literature of usability research, quite a number of empirical studies on UIM and ROI have been documented but the results thereof are rather inconclusive. In contrast, RUE and ScM are more recent approaches, especially the latter. The authors will present alternative views and new findings on these topics to stimulate the participants to further ponder the issues. The fifth contribution addresses usability with a relatively basic approach – how some personality dimensions influence the interaction between the user's mental models and the system's user interface; some implications for usability evaluation may be drawn.

- **Group 2: Web Usability & Accessibility Guidelines and Tools** – Three interesting contributions address this specific theme with different focuses and approaches. In the first contribution, the authors adopt a review method to classify, compare and contrast some existing quality models for web usability evaluation, and consequently develop a reference framework for quality models in Web automated evaluation. In the second contribution, the authors tackle the topic with a formal approach via the use of Guideline Definition Language that enables automated evaluation of websites. Further, quality models involved in such an automated evaluation are defined and exemplified. In the third contribution, the authors report some preliminary findings on the application of a newly developed tool ADIS (Accessibility and Device Independence System), which enables the automatic and dynamic generation of accessible and device-independent versions of websites. ADIS is still work in progress and will be finalized later.

- **Group 3: Software Quality Models and Standards** – The four contributions in this group examine the applicability of existing quality models and standards to the design and evaluation of different kinds of software systems. User satisfaction – one of the key usability metrics – is analyzed with reference to some real-life experiences and to Herzberg's theory of job satisfaction as well as Kano's quality model. Implications about 'must-have' and 'attractive' properties of a user interface for user (dis)satisfaction are inferred**.** Further, two contributions commonly review the definition of usability as specified in the widely disseminated standard: ISO 9241-11 but from different angles and with different case studies. One of these two contributions also reviews the adequacy of ISO 9126-1 and ISO 19796-1 to support the design and evaluation of digital libraries. In the fourth contribution, the authors introduce a framework for usability evaluation based on the OlivaNova Model Executive technology (ONME) technology derived from OO-Method, and present the quality model underlying the framework. Such a usability framework is proved useful for identifying usability problems at both the problem space and the solution space level.

- **Group 4: Context of Use and eCommerce** – A common feature of the three contributions in this theme group is their emphasis on the significant role of context of use in user interface quality models. Another common feature is their choice of eCommerce as the application context to illustrate as well as support their arguments. Specifically, one contribution focuses on user interface quality for customer guidance in eCommerce, the other identifies heuristics and guidelines for the design and evaluation of eCommerce, and the third addresses the credibility of eShops in a specific cultural context. In addition, contexts such as mobile ICT and online communities are dealt with in the contributions.

All the contributions have been delivered to at least two peer reviewers, who were required to complete a review report template to describe major strengths and major weaknesses of individual contributions in terms of their originality, relevance, validity, and presentation, and to propose improvement suggestions.

Structure of the Workshop

On the first day of the Workshop, four groups of presentations will take place with each of them consisting of a 15-minute talk and a 5-minute questioning period. Towards the end of the day, there will be a break-out session: the participants will be divided into four groups to identify specific questions for each of the respective groups. Discussion will be guided and moderated by group leaders.

The second day of the workshop will start with another break out session: the participants will be divided into the same four groups to discuss plausible answers to group-specific questions identified in the previous break-out session. Major ideas being emerged from the group discussions will be reported back to the plenary by the group leaders. Plans for future work will then be inferred.

Outcomes of the Workshop

The contributions and the results of the group discussions will orient the future work on the research inquiries pertaining to the overarching theme of the Workshop - User Interface Quality Models. Drafting of a white paper based on the ideas thus garnered will be implemented. The white paper will be circulated to the interested Workshop participants for further refinement. The white paper to be finalized by the Workshop co-chairs will become the introducing chapter of a book, which will be planned to be submitted to the Springer-Kluwer International Series in HCI.

Jean Vanderdonckt
Effie Lai-Chong Law
Ebba Thora Hvannberg

30th August 2005

TABLE OF CONTENT

Usability Inspection Methods: Do We Ask the Right Questions?

David A. Carr
Department of Computer Science and Electrical Engineering
Luleå University of Technology
SE-971 87 Luleå, Sweden
David.Carr@ltu.se
+46 920 491 965

ABSTRACT
Usability inspection methods are widely used, and often, they are the only methods used. Despite their popularity, there are serious questions about their effectiveness and validity. Many researchers question the quality of the problems found by inspection methods. Consensus is that inspection methods tend to find superficial problems and that they tend to miss problems with interaction and task support. Unfortunately, the very problems that inspection methods miss tend to be those that are most severe for users. This position paper argues that usability inspection methods concentrate on the wrong features of the application and ignore issues such as user task support. If usability inspection methods are to give better input into the development process, they must concentrate on aspects of quality other than surface features. Early in development, suitability to user tasks is the most important aspect of quality. Therefore, inspection methods aimed at early stage development must concentrate on this issue.

Author Keywords
Usability evaluation, usability inspection methods

ACM Classification Keywords
H5.2. User Interfaces – Evaluation/Methodology

INTRODUCTION

During the last 15 years, usability inspection methods have enjoyed increasing popularity. They are seen as a lower-cost alternative to tests with users. Inspection method proponents claim advantages over user testing, such as those for Heuristic Evaluation [9, 11]:

- They are employable at an earlier stage than tests with users are, as they may be used on design sketches.
- They produce good results for inexperienced designers.
- They are more cost effective than tests with users are.
- They are complimentary to tests with users in that they find different usability problems.

However, these advantages have been called into question. Reasons to doubt these advantages include:

- User testing methods such as Cooperative Evaluation [13] *can* be employed at an early stage.
- A clear link has been shown between evaluator expertise and results [6, 10].
- It has been shown that the inspection methods produce many false indications of usability problems, calling into question their cost effectiveness [3].
- Most importantly, many of the problems found are generally criticized as being minor or superficial, and thus, they divert resources from discovering and fixing more serious problems [2].

In attempts to improve inspection methods and overcome criticism, a number of different inspection methods and a number of modifications to existing methods have been proposed. Problems persist however. One possible cause can be the mismatch between the user's view of the tasks to be carried out, and how the system demands that the tasks be performed. Indeed, the ISO 9126 standard for software quality recognizes suitability as a quality subcharacteristic. While ISO 9126 classifies suitability under functionality, a related standard, ISO 9241-11, defines usability as:

> "… the effectiveness, efficiency, and satisfaction with which specified users can achieve specified goals in particular environments …",

which clearly includes a suitability aspect under "specified goals".

This paper explores one possible cause for the problems. Inspection methods do not examine the suitability of the system to the user's task. Instead they focus on narrow issues of use such as graphic design, terminology, or support for learning. Quite simply, current inspection methods ask the wrong questions.

The remainder of this paper gives a brief overview of usability inspection methods. It then analyzes two common methods, Cognitive Walkthrough and Heuristic Evaluation, in the context of two web-based airline reservation systems. It shows how serious usability problems could be missed by these methods. Finally, it concludes with a proposal that early-stage inspection concentrate on suitability rather than other aspects.

BACKGROUND

There are many different usability inspection methods. They all share the same basic format: A computer system or prototype is evaluated by inspection with respect to an evaluation procedure and associated inspection criteria. The evaluators are not expected to be actual users of the

system, but may be designers, implementers, or usability experts. Inspection methods are commonly classified with regard to the following major properties:

- *Individual inspection versus group inspection*. An evaluator working alone can carry out the inspection, or a group of evaluators can work together to create a common inspection. An inspection method that merges the results of individuals working alone is considered an individual inspection method.
- *Guided versus unguided inspection*. For a guided inspection, a set of use-scenarios or user tasks is prepared in advance, and evaluators attempt to use the system to carry out the scenarios or tasks. Unguided inspections rely on the evaluators to determine how the inspection will be organized. Quite often, evaluators simply leaf through system displays without regard to dialog sequence.
- *Level of expertise*. Some methods are designed for evaluators with high usability-related experience. Others claim that they may be successfully used by anyone.
- *Rule/Guideline set.* Each method has its own set of usability principles that guide the inspection. Variations or new methods can be constructed by modifying the principles.

The two most popular inspection methods are Cognitive Walkthrough [18] and Heuristic Evaluation [11]:

Cognitive Walkthrough is a guided, group-inspection technique. Its creators recommend that at least one evaluator in the group be experienced in cognitive science. So, it must be classified as requiring a high level of expertise. The rule set for Cognitive Walkthrough focuses on whether or not a user can learn the application while using it. The rule set has undergone several revisions. Originally, it was complex, encompassing several pages. This was simplified and reduced to four questions in the version presented in [18]. Even these four questions have also been criticized as too elaborate, and several research groups have tried to shorten it. For example, Spencer reduces the set to two questions [17]. The Cognitive Walkthrough method specifies that evaluators be provided with a set of solved tasks as a guide. The fact that evaluators are told how the designers envision the tasks will be done appears to inhibit evaluators from applying their own experience [3]. This leads them to miss problems that they might have discovered if they had tried to solve the tasks themselves.

Sears and Hess [16] demonstrated the importance of level of detail in task descriptions for Cognitive Walkthroughs. They conducted a Cognitive Walkthrough where half of the evaluators were provided with short task descriptions, and the rest were provided with the short descriptions plus step-by-step instructions on how to accomplish the task. There were significant differences in problem reports between the two groups. Evaluators with step-by-step instructions reported significantly fewer problems of the type "users will not be able to find the control" than evaluators using short descriptions. Evaluators with step-by-step instructions also reported significantly more problems related to user knowledge, inappropriate feedback, and system failures than those with short descriptions.

Heuristic Evaluation is an unguided, individual inspection method. Heuristic Evaluation tries to increase its thoroughness (real problems found divided by real problems that exist), by combining the results of multiple individual evaluations. Heuristic Evaluation's creators claim that anyone can use it, and no prior expertise is required. However, they have also published a study showing improved results with experts [10]. They conclude that evaluators who are both experienced usability analysts and experienced in the problem domain are best. Heuristic Evaluation provides 10 usability heuristics as a rule set. As with Cognitive Walkthrough, these rules have varied over time. The final set developed by the method's creators was published in [11]. Other research groups have suggested modifications to the rule set as a way of improving it. Muller and McClard [14] suggest adding three heuristics extracted from the principles of participatory design. These new heuristics are intended to emphasize work quality and satisfaction issues. Law and Hvannberg [12] suggest replacing the heuristics with Gerhardt-Powals' cognitive engineering principles as a way to ground Heuristic Evaluation on well-recognized principles and improve its effectiveness. Similarly, Hornbæk and Frøkjær [7] propose using the metaphors of human thought in order to get evaluators to think beyond the details of the interface.

The principle criticism against Heuristic Evaluation is that it produces a high number of false alarms (usability problems that are not actually usability problems). In a large study of the effectiveness of Heuristic Evaluation, Cockton and Woolrych [2] found that 65% of all predicted problems did not occur in actual use. Half of the false alarms were based on an incorrect understanding of the system's operation or an incorrect understanding of Human-Computer Interaction principles (bogus problems). Furthermore, evaluators missed 43% of severe problems, and they were most likely to miss "constructable" problems, i. e. those requiring several interaction steps to find.

Researchers have taken two approaches to reducing the high false alarm rate. Sears [15] proposes a hybrid technique, the Heuristic Walkthrough, which employs a Cognitive Walkthrough first in order to limit the scope of the Heuristic Evaluation. However, Cockton, et. al. [3] criticize this approach because Cognitive Walkthrough is known to have low thoroughness.

The Discovery and Analysis Resources (DARe) model [4] suggests a different approach. DARe divides the evaluation into two phases. The first phase is a search for possible problems, and the second is a problem confirmation (reduction) phase. By instituting a structured problem reporting form intended to assist in understanding evaluator behavior, DARe's creators also got a dramatic drop in false alarms. The reporting form encouraged the evaluators to more carefully apply the heuristics. The DARe model differentiates four different discover resources [4]:

- *System Scanning:* looking at the system without a strategy.
- *System Searching:* looking at the system systematically.
- *Goal Playing:* incorporates specific user goals such as finding a piece of information.
- *Method Following:* walking through a preconceive method to complete user tasks.

The structured problem reporting form allows analysis of the discovery resources used, and Cockton, et. al. [5] report about system scanning:

> "For too many analysts, this is the limit of forethought and planning during inspection."

The DARe reporting form was extended and has lead to the identification of seven knowledge resources used in usability inspection [19]. These are:

- knowledge of users and their abilities,
- task knowledge,
- application domain knowledge,
- interaction knowledge,
- product knowledge,
- design knowledge, and
- technical (system implementation) knowledge

An analysis of 51 problem reports in a Heuristic Evaluation by groups of final year Human-Computer Interaction students shows that task knowledge was used in only 4 of 190 cases [19].

Desurvire [6] reported on comparisons of Cognitive Walkthroughs, Heuristic Evaluations, and experimental evaluations with users. She analyzed the results with respect to the evaluator's work role: human-factors expert, software developer, and non-expert. Experts performed the best, but had low thoroughness with mean rates of 44% with Heuristic Evaluation and 28% with Cognitive Walkthrough when compared to problems actually found in laboratory testing. Thoroughness rates for other roles were at best half of expert performance. The problems were classified as to origin including those originating from task support. On task related problems, expert thoroughness dropped to 11% for Heuristic Evaluation and 12% for Cognitive Walkthrough. Thoroughness for task-related problems in other roles was essentially zero.

AN EXAMPLE

As an example in this paper, let us consider two, web-based, airline ticket systems. All major airlines currently provide such a system. The systems are designed to allow the traveler to bypass the travel agent by reserving a seat and purchasing a ticket directly from the airline. All such systems seem to be remarkably similar. They are organized as a chain of forms in what is known as the "wizard" style.

The example will be restricted to one scenario. However, it will be a realistic, but not a trivial one. In our scenario, the user wants to travel from her current home in L to her parents' home in B. In addition, she would like to attend a conference in V. She would like to return directly home (to L), at the end of the conference. Thus, our user wants to book a triangle flight from L to B to V and back to L (Figure 1).

Current airline pricing policies allow only selling triangle routes at full price. However, our user is aware that she can avoid this price penalty by combining together two or more round-trip tickets into an acceptable itinerary.

ANALYSIS

In order to analyze the strengths and weaknesses of usability inspection methods, we will discuss some problems that occurred when trying to use two airline sites to construct our example itinerary. After a short discussion of the problem, we will analyze whether it is likely that the usability inspection methods would have found the problem. Our belief is that each problem discussed below is serious, and that in several cases, the problem actually undermines the credibility of the web site.

Problem 1 – Poor Task Support

Despite the fact that both Airline 1 and Airline 2 sell triangle tickets, it is not possible to reserve a triangle itinerary on-line. Furthermore, Airline 1 appears to have made every effort to hide their reservations telephone number. Only someone wanting to prove that it is available could be expected to be persistent enough to find it. The reservations number for Airline 2 is available with one-click from a link at the top of each web page. However, the text is light blue and may not be readily perceived.

Even if we consider the alternate of combining together separate round trips, the task is still not well supported. The fundamental problem is that both sites force the user to commit to each round trip independently of the others. Thus, the user must commit to paying for one ticket before she is assured that the second ticket is available. When pricing is considered the problem becomes more complex.

We must now ask if any of our inspection methods could reasonably be expected to expose this problem. First, consider Cognitive Walkthrough: It requires a set of action sequences, and given that they are derived from the above use-scenario, the evaluator might discover the problem. However, it is likely that this discovery is incidental to the Cognitive Walkthrough. Cognitive Walkthrough centers on learning the interface during use and focuses on a

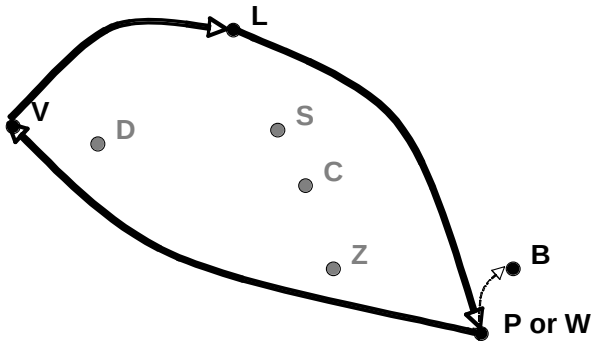


Figure 1. User's goal with intermediate hubs in gray and required ground links shown with dashed lines

step-by-step evaluation of the interface to determine if users could be expected to successfully complete each step. Since it is possible to create a series of steps to complete the task, evaluators might not realize that the uncertainties of premature commitment are a serious problem, especially when evaluators are focused on learning. We must also remember Desurvire's results [6], which show extremely low (12%) success in finding task related problems that are later exposed with user testing.

Next, let us consider Heuristic Evaluation: Heuristic Evaluation does not require evaluators to actually consider tasks, and the literature reports evaluators rarely do so (2% use of task knowledge in [19], 11% thoroughness [6], and analysts most likely to miss problems requiring several steps [2]). So, it seems unlikely that the evaluator will find that our task is not well supported. We can expect a diligent evaluator to note that the contact information is not readily visible for Airline 2; however, this seems incidental as the heuristics don't emphasize graphic design. In fact, we might expect the evaluator to completely miss the difficulty in finding the reservation phone number for Airline 1. It seems a long stretch from "User control and freedom – emergency exits" to a telephone number. This is especially true of inexperienced evaluators. Even Instone's discussion of applying the heuristics to web sites emphasizes web-centered functions such as links to the site home page as an emergency exit [8].

Both inspection methods fail because they don't provide adequate discovery resources [4]. If we consider inspections as especially useful early in the design phase, we can see that focus on user tasks is critical. Bradford [1] emphasizes this in her discussion of the importance of use scenarios when evaluating a high-level design. My experience with students confirms this. Beginning designers do not adequately study the context of use and tend to design for unrealistically simple scenarios. What is really needed at an early stage is an inspection of the design with respect to the context of its use, not with respect to its graphics design and dialog design principles.

Problem 2 – More Segments Give Lower Cost (Airline 1)

The route structure of Airline 1 is such that travel from L to B must go through hub S and can optionally pass through hub C. In addition, B itself has no direct air service. A bus or train must be taken there from either P or W. Airline 1 flies to V (via a partner airline), from both hub C and hub S. (See Figure 2.) Since the flight to V is transcontinental, all departures are midmorning, and connection timing makes it reasonable to travel to the hub the night before departure to V. Therefore, our traveler's task becomes one of scheduling a round-trip from L to P or W and scheduling a trip from S or C to V that interrupts the return trip to L.

Because of the time change between V and Europe, the link between S and V is preferred to the link between C and V. The user's plan would be to fly from P (or W) to S, spend the night in S, and travel to V the next morning. However after entering the S-V link into the web site, the user discovers that all low-priced flights from S go via C. Therefore it seems reasonable that she should fly C-V instead and save the price and inconvenience of flying S-C-V. However, the web site returns only C-S-V connections as low-priced options from C. This Catch-22 situation tends to undermine the user's confidence in the entire web site.

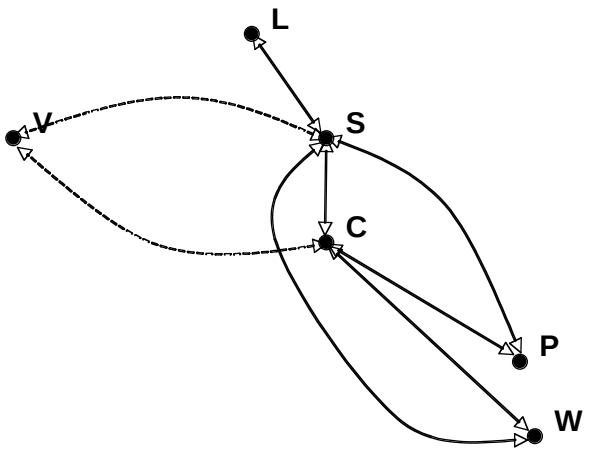


Figure 2. Airline 1's routes.

Now, it can be argued that this problem is a direct result of the inadequate task support discussed above in combination with ticket pricing policies that are outside of control by the user interface. However, it is quite possible that the problem will occur when constructing a single round trip (e.g., flying from P to V with an overnight stay in S or C), and the airline's management should consider the consequences of offering cheaper fares to passengers who are forced to travel unnecessary segments.

As in problem 1, it seems unreasonable to expect evaluators to discover this problem unless they are guided by a suitably complex task set. In addition, the focus of both Cognitive Walkthrough and Heuristic Evaluation is on aspects other than the quality of the results. Therefore, discovery of the Catch-22 situation can be expected to be incidental to the use of either method and primarily dependent on the evaluator's skill in constructing user tasks.

Problem 3 – Bait and Switch (Airline 2)

Airline 2 flies from W to V on two different routes. The first contains two legs, and the second contains three legs with an additional change at Z. The final segment on both routes is the same. (See Figure 3.)

When our user tried Airline 2's web site, the first segment of the outward bound, two-leg route was not available at the indicated low price. The web site indicated this by graying out the segment and displaying a message that the given price was not available on the grayed segment. However, the web site is not being forthright. The indicated low price is not available at all. Selecting any com-

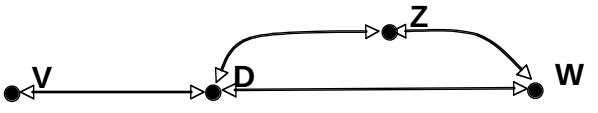


Figure 3. Airline 2's routes.

bination of one of the two outward-bound routes with one of the two return routes results in a fare quote of three or four times the originally quoted fare when the user proceeds to the payment step. The system also displays the following error message.

> "Attention: The total price has changed. The reason might be changed altered taxes/charges or a chosen fare that is not applicable."

While the user may be alerted to the change in price, the fact is that the web site exhibits the behavior of the classic "bait-and-switch" retail fraud.

If we again analyze the possible results of a Cognitive Walkthrough, we are forced to conclude that this problem would be incidentally discovered given two preconditions. First, the evaluator must use a task that includes a route with more than one option. Second, the database must be in a state where one leg is fully booked in the low-price category. Given these conditions, an evaluator will discover that the user can perform the task at a price that is many times the price that is first quoted. However, a plausible scenario can also be constructed where the evaluator fails to notice the magnitude of the price increase.

Similar conclusions can be reached about evaluators using Heuristic Evaluation. With the proper task and database state, evaluators can be expected to notice the price increase. We can also expect evaluators to flag the error message as a violation of the heuristic "Help users, recognize, diagnose, and recover from errors – precise indication of the problem". However, this misses the point and reports a minor error when the real problem is very serious.

DISCUSSION

As we can see from the above, both Cognitive Walkthrough and Heuristic Evaluation can miss serious problems. In the examples above, this depends on the degree to which:

- evaluators follow non-trivial user tasks as they evaluate the interface, and
- evaluators apply their own experience.

Constructing reasonable discovery resources is an area where all current usability inspection methods could be improved. Published resources tend to be sketchy about what is required. In the author's experience, inexperienced designers have problems in this area. The work of Cockton and his group [2, 3, 4, 5, 19] demonstrates that evaluators rarely apply task knowledge or methodically follow use scenarios. Finally, the example web applications show that even large companies with significant resources have problems with constructing realistic scenarios of use. Therefore, it seems that two remedies need to be undertaken:

- An inspection method that checks the results of the requirements and user-task analyses needs to be developed and tested. This method could be independent of any usability evaluation method. The method sketched by Bradford [1] could serve as a basis for further development.
- Existing methods must have their discovery resources strengthened, and their documentation must emphasize the importance of realistic user tasks. Methods without adequate emphasis on preparation and rigor should be discarded as worthless.

The issue of evaluators applying their own experience is a delicate one. On the one hand, it has been shown that experienced evaluators get better results. On the other hand, allowing evaluators a free reign can result in reports of problems by the evaluators due to style or philosophical issues rather than actual usability problems. It seems that using a detailed problem report such as in [4] would mitigate these problems. At the same time, experienced evaluators should be allowed to report problems outside of the confines of the inspection method so long as they are clearly labeled.

CONCLUSION

Usability inspection methods are here to stay. Their promise of quick results at little cost is too attractive, despite their shortcomings. Therefore, these techniques need to be improved so that they report only real problems with the added guarantee that *nearly all* severe problems are detected. Since current methods often miss problems that are a result of an interaction sequence, evaluators must be encouraged to employ realistic interaction sequences during inspection. In order to encourage realistic interaction sequences, inspection methods must be redesigned so that the results of user-requirement and user-task analyses are transferred into the method. In addition, there must be an evaluation of the quality of these analyses. Only in this way will inexperienced designers and evaluators be guided to execute meaningful usability inspections.

ACKNOWLEDGEMENT

Thanks go to Carl Rollo for proofreading drafts of this paper. His efforts markedly improved its readability.

REFERENCES

1. Bradford, J. S., Evaluating high-level design: synergistic use of inspection and usability methods for evaluating early software designs, in *Usability Inspection Methods*, Nielsen, J. and Mack, R., eds., John Wiley and Sons, 1994, ISBN 0-471-01877-5, 235-253.
2. Cockton, G. and Woolrych, A., Understanding inspection methods: lessons from an assessment of heuristic evaluation, *Proceedings of IHM-HCI 2001*, 171-192.
3. Cockton, G., Lavey, D., and Woolrych, A., Inspection-based methods, in *The Human-Computer Interaction Handbook*, Jacko, J. A. and Sears A., eds., Lawrence Erlbaum Associates, 2003, ISBN 0-8058-3838-4, 1118-1138.
4. Cockton, G., Woolrych, A., Hall, L. and Hindmarch, M., Changing analysts' tunes: the surprising impact of a new instrument for usability inspection method assessment, *Proceedings of HCI 2003*, 145-162.

5. Cockton, G., Woolrych, A. and Hindmarch, M., Reconditioned merchandise: extended structure report formats in usability inspection, *CHI 2004 Conference Companion*, 1433-1436.
6. Desurvire, H. W., Faster, cheaper!! Are usability methods as effective as empirical testing?, in *Usability Inspection Methods*, Nielsen, J. and Mack, R., eds., John Wiley and Sons, 1994, ISBN 0-471-01877-5, 173-202.
7. Hornbæk, K. and Frøkjær, E., Evaluating user interfaces with metaphors of human thinking, *Proc. of User Interfaces for All 2003*, Springer LNCS 2615, 486-507.
8. Instone, K., Site usability heuristics for the web, http:// user-experience.org/uefiles/writings/heuristics.html, visited 6 June 2005.
9. Nielsen, J., and Molich, R., Heuristic evaluation of user interfaces, *Proceedings of CHI'90*, 249-256
10. Nielsen, J., Finding usability problems through heuristic evaluation, *Proceedings of CHI'92*, 373-380.
11. Nielsen, J,. Heuristic evaluation, in *Usability Inspection Methods*, Nielsen, J. and Mack, R., eds., John Wiley and Sons, 1994, ISBN 0-471-01877-5, 25-62.
12. Law, L.-C. and Hvannberg, E.T., Analysis of strategies for estimating and improving effectiveness of heuristic evaluation, *Proceedings of NordiCHI 2004*, 241-250.
13. Monk, A., Wright, P., Haber, J., and Davenport, L., Improving Your Human-Computer Interface: A Practical Technique, Prentice Hall, 1993, ISBN 0-13-010034.
14. Muller, M. J. and McClard, A., Validating an extension to participatory heuristic evaluation: quality of work and quality of work life, *CHI 1995 Conference Companion*, 115-116.
15. Sears, A., Heuristic walkthroughs: finding the problems without the noise, *International Journal of Human-Computer Interaction*, 9(3), 1997, 213-234.
16. Sears, A., and Hess, D. J., The effect of task description detail on evaluator performance with cognitive walkthroughs, *CHI 98 Conference Companion*, 259-260.
17. Spencer, R., The streamlined cognitive walkthrough method, working around social constraints encountered in a software development company, *Proceedings of CHI 2000*, 353-359.
18. Wharton, C., Rieman, J., Lewis, C., and Polson, P., The cognitive walkthrough method: a practitioner's guide, in *Usability Inspection Methods*, Nielsen, J. and Mack, R., eds., John Wiley and Sons, 1994, ISBN 0-471-01877-5, 105-140.
19. Woolrych, A., Cockton, G. and Hindmarch, M., Knowledge resources in usability inspection, to appear in *Proceedings of HCI 2005*.

Benefits of usability work – does it pay off?

Tobias Uldall-Espersen
Datalogisk Institut, Københavns Universitet
Universitetsparken 1, DK-2100 Copenhagen Denmark
tobiasue@diku.dk

ABSTRACT
The value of work to improve the usability of information systems in industrial settings is rarely accounted for. This study reports from an experiment where an administration and risk management system in a bank were improved through three versions during a period of six month. The system has ten users and usability data was collected through questionnaires and logging of data from practical use. The experiment shows how it was possible to improve the system over a broad range of measures covering efficiency, effectiveness and user satisfaction issues. The analysis of return on investment (ROI) of this usability work shows a pay back time of five years, but in this case, the most important improvements were found in issues not accounted for by the ROI. These issues, e.g. expected increased use of the system and a postponement of a replacement of the system alone justified the usability work. This indicates how return on investment analyses are at risk of missing the most important issues.

Author Keywords
Field Experiment, Case study, HCI, Usability engineering, Usability factors, Evaluation Techniques, ISO 9241-11, Return on investment, Cost-Benefits, Metaphors of Human Thinking, Think Aloud, Questionnaire, Logging of usage date, Business process reengineering

ACM Classification Keywords

H.5.2. User Interfaces, D.2. Software Engineering, D.2.9. Management

INTRODUCTION

Traditionally in software development, a number of processes have special focus, e.g. requirement definition, design specification, implementation and test. These processes must be completed and documented consistently and systematically, but this has not sufficiently ensured usability of the developed systems. If usability is to be ensured, the software development processes must be enhanced with further activities. There is a diversity of different techniques, which has been diligently compared, but there seems to be a lack of field studies, documenting the impact of usability work in industrial settings. The purpose of this study is to report how usability has been improved and documented in an industrial software engineering experiment.

The experiment was designed and conducted with the aim of the highest possible realism. All data is collected from real users performing real tasks using a real system in a specific business. Data is based on the usage of an information system used by ten users in a bank in Denmark. Using three versions of the system, data has been collected through six month of logging and two questionnaire surveys have been conducted in the start and at the end of the experiment. The usability work is made by the software engineer who has build the system, which could be a useful approach in future projects.

AIM OF THE EXPERIMENT

The aim of the experiment was to investigate the following:

- Can usability of the system be improved?
- Can changes in usability be identified, documented and measured?
- Does it pay off to identify usability issues and to improve the system?

Can the level of usability be measured, and changes in usability between versions of the software be identified, documented and explained? In the current experiment, this is a precondition for the objective evaluation of the benefits, just as it is necessary to document product performance in an industrial context.

THE SYSTEM AND THE USERS

The evaluated system is a small MS Windows based information system, which was previously developed by the author of this paper. It was implemented in the PowerBuilder programming language and was connected to an Informix database system. The system was used in the bank a couple of years before the experiment was conducted. It is used for administrative purposes, for reporting and for management of risk in relation to a specialized loan department with a total loan amount of about 150 millions euro. The system consists of 10-12 primary windows, where data can be searched, inserted and updated. Further, it has a number of secondary windows and ten reports. The system is the users primary tool for the administration of the loans. In the experiment, only the primary windows were evaluated.

During the test-period ten users had access to the system, and all of them contributed with data. The users were all bank employees with a financial education, and they worked in two different departments. Eight of the users came from the primary department (the primary users), where the system was used mostly. Two secondary users came from a department, which worked with control and risk management for the overall company group. In both departments, the distribution of gender was equal. At the

beginning of the experiment the newest employee had been in the department for about 9 month, and hence all of the users where used to internal routines and business rules. All of the primary users had prior to the experiment access to the system, one of them used it only sparsely. Of the secondary users one used the system sparsely and the other got access to the system when the experiment started. The nine users who had access to the system prior to the experiment, all participated in the survey in the start and at the end. They were asked to express their experience about Information Technology (IT) and about the system. Four respondents had middle experience with IT, two had little, and the last three had very great, great, and very little IT experience. Overall a level of experience just below middle. Regarding use of the system, five respondents had much experience, one had middle, two had little and one had very little. The total experience with the system was a little higher than middle, but the distribution was very unequal.

EXPERIMENTAL METHOD

To define usability the ISO 9241-11 standard was used:

"Usability: 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."

As suggested by Frøkjær et al. [3] effectiveness, efficiency and satisfaction were a priori considered as being independent factors. Thus, it was necessary to measure all three factors, in order to be able to assess the total change in usability.

The experiment was conducted as a field experiment in an effort to increase its realism compared to laboratory-style studies [8]. During the experiment, two iterations of usability evaluations and system improvements were conducted, and the usage of the initial version as well as the two improved versions were studied. The users worked with all three different versions of the system during the experiment, and all users used the same version at the same time. Every version was used no less than 1½ month. Note that data in this article are limited to the study of the first and the last tested version.

Identification and Processing of Usability Problems

The first part of the experiment consisted in identifying and rectifying potential usability problems. The work was focused with the survey in the start of the experiment, where the users had opportunity to evaluate and comment different parts of the system. The survey was followed by a total of nine usability inspections using both Metaphors of Human Thinking (MOT) based evaluations and think aloud (TA) tests. The MOT evaluation technique is an analytical technique based on five recognized aspects of the human way of thinking [5]. The author conducted two evaluations alone and additionally two pair evaluations with Erik Frøkjær and Kasper Hornbæk – the originators of the MOT-technique. The TA activities were based on the descriptions of Contextual Inquiry [1] pp. 41-66 and TA test [9] pp. 100-115.

The first iteration included the initial questionnaire survey, one MOT based evaluation conducted by the author and three think aloud tests. The survey and the MOT evaluation were link close together, since issues from the survey were given special attention during the MOT evaluation. Both low ratings of parts of the system and the processes as well as user comments were used to point out special issues. This seemed to be an efficient way to bring in some contextual knowledge about the use of the system into the evaluation process. The three TA tests were conducted with three different users, who together used all essential parts of the system. Data from the survey was used for the selection of users and tasks, and the users were ask to collect about two hours work covering the identified tasks, which could be done during the TA sessions.

In the second iteration, two additional users were observed using the improved system for the first time. This served as a special opportunity for evaluating the implemented changes, but the value of the evaluations was limited. More effort should have been put in to identifying tasks covering the improved parts of the system. The iteration was finalized with three MOT evaluations; the first conducted by the author alone and the two following as pair evaluations. All of the nine evaluations were conducted in a very informal way and time was used on both identifying problems and discussing possible solutions.

During the two iterations, 180 intermediate problems were identified. They were consolidated into 99 unique problems from which 40 points of improvement were proposed. The consolidation process was conducted with the main objective to remove clearly identically problems, e.g. problems of the same type regarding the same objects observed during the same processes. Because the author had observed all intermediate problems except 13 originating from the questionnaires, the consolidation could be done without much difficulty, as the context of the problems was clear. The purpose of the proposed points of improvement was to group the consolidated problems into collections that could be fixed together through one proposed redesign. For every point of improvement a possible redesign was worked out and shortly described, and every redesign were then subsequently estimated and informally prioritized from the following four factors:

- How serious are the covered problems?
- How often do they occur?
- How many users do they affect?
- How much time is needed to implement the solution?

Based on the priority, a plan for implementation was developed. According to the goal of optimizing the return on investment (ROI), the focus was on solving the issues, which gave the users most value for money. At this stage of the usability improvement process, it seemed appropriate to be guided by suppositions about ROI, but it seemed equally important to let broader strategic considerations guide the selection. For example, one of the most expensive improvements of the system was a

total reengineering of the user manual and help system. As a single activity this would never pay off in the experiment, but on a longer term it increased the value of the system because the users felt more secure when using the system and the system was easier to deploy in another department, which later was done.

In parallel with the traditionally usability engineering activities, the users had access to a database, where all problems were registered and problems could be added and commented. Additionally the use of the system help function was registered. The users were asked after every access to the help function if they found what they were looking for. None of these techniques had much influence on the result, whereas an informal dialog concerning some of the identified problems and solutions was useful. Of the 180 intermediate problems, 107 were solved.

Measuring Usability

The second part of the experiment consisted in measuring the changes of usability, i.e. improvements of efficiency, effectiveness and satisfaction. For this purpose, logging and two questionnaire surveys were used. The logging was done through the entire six-month experimental period, whereas the surveys were conducted in the start and at the end of the period. All logged data originates from real use of the system in the production environment supporting the highest possible realism. The changes in efficiency were measured by logging, user satisfaction by surveys and effectiveness by both logging and surveys.

Hypotheses

To guide the experiment, a set of 25 hypotheses was put forward, which could help identifying changes in usability. To each hypothesis, a description was made of its relevance and how to clarify it. The most important hypotheses are described below.

For efficiency considerations, six hypotheses were put forward, e.g.:

- *Hypothesis:* Applying usability work, one can reduce the orientation time, i.e. the time from a window is activated until the user interacts with it.

  Relevance: A reduction in orientation time indicates that less effort is required for the user to get a general view of a window, to recall the task and to choose required functionality.

  Clarifying: Through logging it would be possible to measure the time from the window is activated to the first following user-generated event.

- *Hypothesis:* Applying usability work, one can reduce the time from a user starts the program to the first useful windows (e.g. a window with data relevant to the user) is shown.

  Relevance: A reduction in startup time indicates that less effort is required for the user to recall the use of the system.

  Clarifying: Through logging it would be possible to measure the time from the system start event to the activation of the first window with user selected data.

For effectiveness considerations, 12 hypotheses were put forward, e.g.:

- *Hypothesis:* Applying usability work, one can reduce the number of searches for and updates of the same data object.

  Relevance: Repeated search for and updates of the same data object could indicate bad effectiveness e.g. that the user is insecure or makes erroneous updates.

  Clarifying: Through logging, count the number of searches for and updates of each data object.

- *Hypothesis:* Applying usability work, one can reduce the number of interruptions the user experiences.

  Relevance: Each interruption could distract and annoy the user.

  Clarifying: Through logging, count the number of interruptions in each window.

- *Hypothesis:* Applying usability work, one can reduce the number of unsaved changes of data.

  Relevance: Canceled changes indicate that the user starts modifying an object that should not be modified.

  Clarifying: Through logging, count the number of canceled changes in each session.

For satisfaction considerations, seven hypotheses were put forward, e.g.:

- *Hypothesis:* Applying usability work, one can improve the users satisfaction with the system.

  Relevance: Unsatisfied users could be a usability issue.

  Clarifying: Ask the users about their satisfaction in the survey.

- *Hypothesis:* Applying usability work, it is possible to increase the users perception of working efficiently.

  Relevance: Feeling not wasting time could increase user satisfaction.

  Clarifying: Through the survey, ask the user questions about task solving experience.

- *Hypothesis:* Applying usability work, the system can be made more convenient to use.

  Relevance: An inconvenient system could reduce user satisfaction.

  Clarifying: Through the survey, ask the user questions about matters of inconveniency, irritations etc.

Logging

The system logged 31 different types of events generated by the users and their usage of the system, e.g. opening/closing of a window, click with the mouse, change of focus, changes in data and transactions with the database. In the log, it was possible to identify every data object a user viewed or changed.

Questionnaire

The users were asked for background information (age, gender, …), 19 questions about their satisfaction with the system and 20 questions about their experience with the use of the system. All the 39 questions were answered on

a five-point Likert scale, and each had a text field for free comments where the users were encouraged to go into details in their answers. The 20 questions about their experience were:

How much do you agree in the following questions?

A1. There are tasks in the system, which are difficult to solve.

A2. There are tasks in the system, which are to time consuming to solve.

A3. There are tasks the system should be able to solve, which cannot be solved.

A4. There are tasks in the system I do not know how to solve.

A5. There are special tasks in the system, which I hand over to others.

A6. There are special tasks in the system, which often are handed over to me.

A7. There are tasks in the system, which I often need help from other in the department to solve.

A8. There are tasks in the system, which I have to help other solving.

A9. There are tasks in the system, which I avoid solving.

A10. There are parts of the system, which annoys me when I use it.

A11. There are parts of the system, which I use without understanding it.

A12. I am insecure about how to solve a task and tries my way.

A13. There are parts of the system, which I feel insecure using.

A14. There are parts of the system, which often give rise to errors.

A15. There are shortcomings in the system.

A16. It is difficult to get around in the system.

A17. I miss feedback from the system.

A18. The system does something different from what I expect.

A19. The system expects me to solve a task in another sequence, than I would have done.

A20. I often make errors in parts of the system.

Seventeen of the questions abut satisfaction dealt with parts of the system and two were general. The non-general questions were:

How satisfied are you with the following parts of the system?

B1. The search function where you can select engagement, investor or guarantor from a list.

B2. Selection of and navigation between windows.

B3. Set up an engagement.

B4. Adding a property.

B5. Adding a tenancy.

B6. Adding a mortgage.

B7. Set up an investor.

B8. Set up a guarantor.

B9. General view of engagements.

B10. General view of properties.

B11. General view of tenancies.

B12. General view of mortgages.

B13. General view of investors.

B14. General view of guarantors.

B15. Reports.

B16. Import of data from data warehouse.

B17. The total general view.

The two general questions were:

B18. How satisfied are you generally with the system?

B19. If a colleague in another department were thinking of starting using the system, would you recommend it?

DATA

About 250.000 log entries originating from about 190 hours of active use of the system were logged. This covers 509 different logins from the users in the period from week 16 to week 44 in year 2003. Totally, the users had been logged on for 1486 hours of which the system was active in 13% of the time. A great deal of the log turned out to be difficult to analyze quantitatively, and was not analyzed. The reason for this was that the usage of the system was more diversified than expected and there were considerable differences in how much the users used the different parts of the system. Nine of the ten users chose to participate in the questionnaire surveys. The surveys were conducted by a third department and were done using a professional web based tool.

In addition to the primary data collection, the use of resources where carefully registered and evaluated, and a final evaluation of the usability improvement process was conducted. To be able to estimate cost-benefits and return on investment later, all time consumption according to relevant activities were collected. This includes time used to study techniques, analyze the system, analyze problems and solutions and the time used to solve the selected problems. The final evaluation was an interview with the responsible department manager, who also was one of the primary users of the system. The purpose of the interview was to get a reaction from the manager to the completed process, the achieved results and the return on investment.

RESULTS

The experiment resulted in important changes of usability in the system and clear changes of efficiency, effectiveness and user satisfaction were documented.

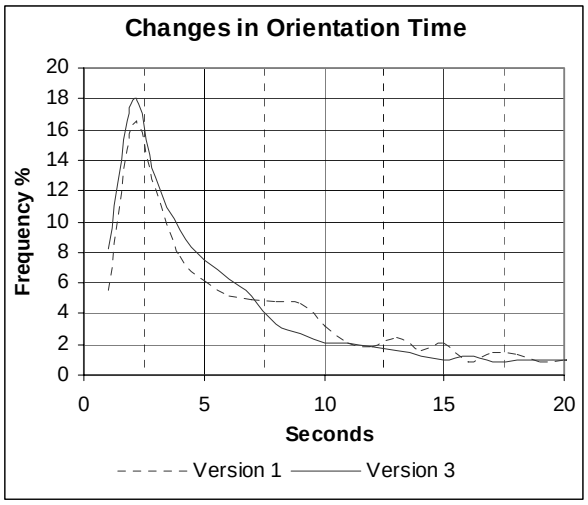


Figure 1 – Distribution of time used to orientation i.e. the time from a window is activated until the user interacts with it. The frequency plot shows a change in efficiency between version 1 and version 3. About 68% of all orientations are done on less than 7.5 second in version 3, which is about 10%-point more than for version 1. In version 1, 28% of the orientations were done between 7.5 and 20 seconds, which is about 7%-point more than for version 3. The plot is based on about 7,800 observations.

Efficiency

The changes in efficiency were primarily obtained through a better adaptation to users work practice and a reduction in time used to startup and to orientation. The experiment uncovered how the users work practice had changed from the first to the final version of the system. Existent routines evolved, new ones developed and the users had specialized in different parts of the system. This caused a need for adaptation and business process reengineering (BPR). This is shown by the following example: Twice a year the users had to print a summary of every engagement (about 150) and of every investor (about 600). This was done by manually selecting one engagement / investor at a time and then pressing the print icon, which was a rather time consuming process. In the final version of the system there was added a 'print all engagements' and a 'print all investors' function, which saved a great deal of time. In a similar manner, a great part of the data maintenance processes was reengineered. For example the yearly process of registering an engagement account was reengineered from registrations in three general windows to registration in one specialized window, also displaying the last registered account information. This made it easy both to register data and to control the typed in numbers against last registration. The experiment resulted in considerable improvement of the adaptation of the users primary work processes to the system. For a number of important work processes, performance was improved by 40%-76%.

In addition, substantial improvements regarding the startup of the system, navigation in the system and orientation in different windows were documented. The time from starting the system to viewing relevant data was reduced from 43 seconds to 31 (28%). The reduction is even more evident, if we watch the 90% percentile: The time has been reduced from 27.4 to 16.9 seconds (38%). This improvement saves time for the users and may even result in further use of the system because of the faster access to data. The database access time did not change significant during the experiment, and the observed effect can to a certain extent be explained by a centralization of the search facilities to one window and some changes in the navigation. In the first version, the users did all navigation through menu icons or menu entries. In the last version, the navigation was done through buttons placed in every window. Technically, slightly more complex to implement, but it seems to be much more usable. From the measured startup and search times, the experiment shows a reduction of time used to navigation in the order of 60%-70%.

If we look at the time used from a window is opened until the user interacts with it (called *orientation time*), another important result has been pointed out. For the 90% percentile of more than 7.800 observations, the time has been reduced from 8.3 to 5.8 seconds (30%). This is an important result; the fact that it is possible to improve the time used to orientation and forming a general view, could reduce the waste of time in the system. Figure 1 shows a frequency plot of the time used to orientation. In version 1, about 58% of the orientations was done on less than 7.5 second and about 28% between 7.5 and 20 seconds. In version 3, the same numbers where about 68% and 21% which shows, that the users uses less time to orientations. The amount of slow observations (time>20 seconds) seems to bee stable. Some external factors might have influenced the orientation time, e.g. changes in the physical environment or the users learning the interface, but no obvious sources of influence exists. During the experiment there were no changes in the physical environment, and there is no indication of a learning effect within each of the time periods where a single version of the system has been used.

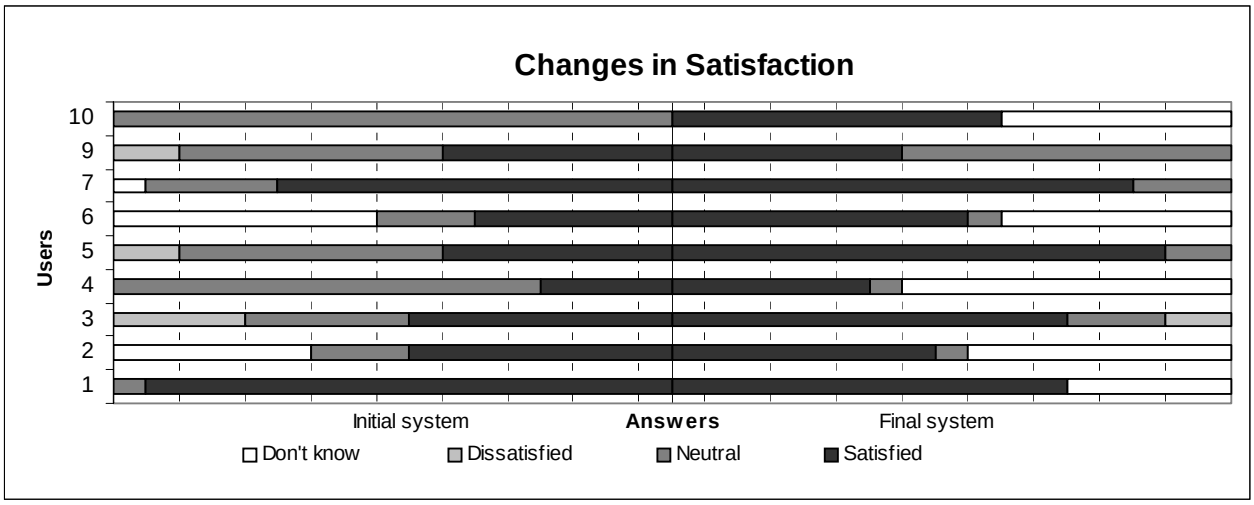


Figure 2 – At the left half is shown the satisfaction with the initial system, and on the right half the satisfaction with the final system. Overall, the users seem less dissatisfied and more satisfied. An increased number of *don't know* could confuse the picture. These have been rejected in the analysis. User 8 did not participate in the questionnaire.

Effectiveness

The changes in effectiveness was documented through both surveys and logging. The users' experience (based on 360 answers from the surveys) with the system did only improve slightly (0.2 points), and only one user expressed a significant change (0.6 point). The improvements documented by the logging were more substantial. A reduction in interruptions on about 88% was observed. To a degree, this fact could be explained by a change in feedback and an improved error handling. The result is fewer disturbances of the users. Likewise, a 79% reduction in the number of times, the users starts changing data without saving were observed. This expresses a more suitable use of the system.

User Satisfaction

The questionnaires gave 138 answers about the initial satisfaction with the system and 128 about the final. On a scale from 1 (very unsatisfied) to 5 (very satisfied) the initial satisfaction was 3.56 and the final was 4.03. In a meta-analysis based on 127 measurements, Nielsen & Levy [11] has shown that 3.5 is a normal level for interfaces before improvement. However, there seems to be considerable variation. On 6 of the 17 questions, there were significant improvements (0.5 – 1 point) and four of nine users expressed significant improvements (0.7 – 0.9 point). Figure 2 shows the change of user satisfaction. Only one user did express dissatisfaction with parts of the improved system. The general questions did not show any significant changes.

Time and Management Consideration

As an important part of the experiment, all relevant time consumption was registered. Overall 15 hours were used to study the techniques, 25 hours to analyze the system, 10 hours to analyze and prioritize problems and solutions and finally 117 hours to implement improvements. This use of time should be seen in lights of limited experience with usability work, but great knowledge about the system.

The final interview with the department manager revealed a general satisfaction with the undertaken process and the achieved results. He emphasized three special issues, to which he felt the work had contributed:

- To give the user a better general view of data.
- To make the system better and easier to use. The typing has been faster and fewer errors are made.
- The users have gained more confidence with the system and they feel less insecure.

The manager expressed that he had difficulty estimating the actual economic benefits. He stressed how the improvement of effectiveness and satisfaction was more important and interesting to him than the gained efficiency.

DISCUSSION

The experiment shows, that important and appreciable improvements of usability can be made through use of existing usability techniques. The fact that the experiment is a field experiment could support its realism, but it may cause that it is hard to generalize from the results [8]. If we look at the improvements two main observations can be made. Waste time can be reduced and work processes can be supported more efficiently e.g. by BPR. This seems to affect all the usability factors, and may be an issue for other systems.

The considered system was originally developed by the users, and the first version served as a prototype. The prototype was prior to the experiment developed further by the author in close cooperation with the users through a couple of iterations. It shows that even when developing software with serious user involvement, important conditions could be overlooked or misunderstood. It also suggest, that the need for BPR will reappear from time to time when the relation between problems, tools and people has changed. A relation Naur has named The Symmetrical Relation [10] p.29.

Should Software Engineers Work With Usability?

Seventy percent of the resources used in the experiment have been used on the implementation of the identified and designed improvements. This makes it appropriate to aim at ensuring usability of the design before the system is implemented. Therefore, usability work and software engineering should be combined in an iterative process. Close integrating of usability work and software development is also described as necessary in e.g. [6]. This could also be a key to weaken the resistance to usability work, which continues to exist among software developers [13], and be a key to support a more holistic understanding and acceptance of the usability field.

In the experiment, a substantial evaluator effect [7] has been established. The evaluations conducted by the author together with two usability experts revealed considerable different problems compared to the problem identified by the author alone. If the software engineers should evaluate their own products, this could be a problem. They are influenced by earlier decisions, known limitations of the tools, and imaginations of the users, surroundings and so on. This could cause a lack of objectivity and may result in that serious problems are overlooked. The software engineers could however strengthen the process of development if these risks are managed. This might be done by involvement of independent experts, by close cooperation with the users and by focusing on the users experience with the system. In some cases, involvement of software engineers could also result in special focus on solvable problems, which could ensure a higher return on investment.

Does it pay off?

The experiment has documented an improvement of the efficiency on about 10% (35-40 hours a year) in relation the total use of the system. This only covers measured improvement and might be bigger, e.g. the effect of the changed *print all* procedure has not been measured, since it only appeared once in the test period. An important question is whether the efficiency improvement together with the changes in effectiveness and satisfaction can justify the costs of the usability work. Totally, the use of time was about 180 hours, which covers studying the techniques (9%), analyzing the system and the possible changes (21%) and traditional software engineering activities (70%). This indicates that the improved efficiency could finance the work over a five-year period, but what pay back time is expected and realistic in industrial settings? Frøkjær & Korsbæk [4] have shown that public information systems tend to have a considerable lifetime and a payback time on five years might not be unrealistic. The value of the effectiveness and the satisfaction is difficult to calculate, but it seems considerable and more important to the company. In the interview, the department manager expressed, that the completed work has extended the lifetime of the system, and a planned replacement has been postponed. Another department has even adopted the system because of the improvements. This is a saving, which immediately seems to have justified the investment.

However, the experiment has shown that we cannot rely on return on investment (ROI) analyses of usability issues. It would be hard, in advance, to calculate ROI because of the complex measurable outcome variables and a ROI analysis might not have been in favor of the conducted work. Afterwards it seems obvious, that it was a reasonable strategic investment, which supports some of the principles discussed by Dray et al. [2]. They argue in favor of using case studies as a tool to document the effect of usability work, in order to convince practitioners, researchers and decision makers.

Are Efficiency, Effectiveness and Satisfaction Equally Important?

In agreement with Frøkjær et al. [3], the experiment has shown that we cannot expect coherence between efficiency, effectiveness and satisfaction. As an illustration from this experiment, the search function was changed significantly. The users got a more transparent search facility with better search options in a centralized search window. This really improved the effectiveness, but the mean search time increased by 15%. The user satisfaction did not chance significantly. This supports the argument, that the three usability parameters must be considered independently and hence that all three parameters must be measured.

The lack of coherence raises another question. We need to control efficiency, effectiveness and satisfaction, but are they necessarily equally important? Many studies focus on efficiency and/or satisfaction, whereas effectiveness seems to be less considered [3]. Does this mean that effectiveness is less important, or could it be due to the fact, that it is harder to measure? In the interview with the department manager, he expressed that in this context effectiveness and user satisfaction were more important to him than efficiency. Would this be a concern, we need to bring in to the work with usability? The priority depends of the context, but it could be important to be aware of, which preferences the users and decision makers have. If the work is guided by these preferences, it may be easier to prioritize the resources and to use them well.

CONCLUSION

The conducted experiment has shown the following:

- It is possible through systematic work to improve efficiency, effectiveness and satisfaction of the system. The combination of analytical and empirical techniques has shown to be very useful, and it seems that the benefits of using expert-reviews justify the extra costs.
- It is possible through systematic work objectively to document changes in efficiency, effectiveness and satisfaction. Questionnaires and logging are useful in this documentation process and complement each other. To limit the use of resources the measurements must be established and focused on the issues of central importance.
- Efficiency improvements were measured and time of return on investment could be estimated to about five years.

- The economic value of the improvements of the effectiveness of the system and the user satisfaction were impossible to make up although the benefits were clearly identified by the direct users and the manager. In the current experiment the usability improvements raised expectations of increased use of the system also in a new department, and a likely replacement of the system was postponed.
- Although the manager recognized the return on investment analysis based on mainly efficiency measures, he made it clear how the identified effectiveness and user satisfaction improvements were the most important result. They alone justified the usability work. This indicates how return on investment analyses are at risk of missing the most important issues.

ACKNOWLEDGMENTS

I thank Alm. Brand Bank, the department manager and the employees for their participation in this experiment; and my colleagues Erik Frøkjær, Kasper Hornbæk and Mie Nørgaard for their contributions. Kasper and Mie have reviewed an early draft of this paper; Kasper and Erik have performed two collaborative MOT-evaluations; and Erik has been supervising the writing of this article and my master thesis [12] from which the experiment originates. This work is part of the USE-project (Usability Evaluation & Software Design) founded by the Danish Research Agency through the NABIIT Programme Committee (Grant no. 2106-04-0022).

REFERENCES

1. Beyer, H. & Holtzblatt, K. *Contextual Design*. Morgan Kaufmann, 1998
2. Dray, S., Karat, C.-M., Rosenberg, D., Siegel, D. and Wixon, D. Is ROI an Effective Approach for persuading Decision-Makers of the Value of User-Centered Design? *Panels CHI 2005*, 1168-1169
3. Frøkjær, E., Hertzum, M. and Hornbæk, K. Measuring Usability: Are Effectiveness, Efficiency, and Satisfaction Really Correlated? *Proc. CHI 2000*, 345-352
4. Frøkjær, E. & Korsbæk, H. (1992). Informatization Policies in Denmark. In Frissen, Bekkers, Brussaard, Snellen, & Wolters (Eds.), *European Public Administration and Informatization* (pp. 25-47). IOS Press.
5. Hornbæk, K. & Frøkjær, E. Evaluating User Interfaces with metaphors of Human Thinking in *User Interfaces for All*, Lecture Notes in Computer Science 2615, Springer-Verlag, 486-507
6. Hornbæk, K. & Frøkjær, E. Comparing Usability Problems and Redesign Proposals as Input to Practical System Development. *Papers CHI 2005*, pp. 391-400
7. Jacobsen, N.E., Hertzum, M. & John, B.E. The Evaluator Effect in Usability Test. *Late-Breaking Results CHI 1998*
8. McGrath, J. Methodology Matters: Doing Research In The Behavioral And Social Sciences, pp. 152-169 in Baecker, R. M.: *Reading in human-computer interaction: toward the year 2000*, Morgan Kaufmann Publishers, Inc., 1995
9. Molich, R. *Brugervenlige edb-systemer*, Ingeniøren Bøger, 1994, 2. edition 2001
10. Naur, P. Computing: A Human Activity. *ACM Press / Addison-Wesley*, 1992
11. Nielsen, J. & Levy, J. Measuring Usability – Preference vs. Performance. *Communications of the ACM, April 1994/ Vol. 37, No. 4*
12. Uldall-Espersen, Tobias, 2004. Benefits of usability work – does it pay of, *Department of Computer Science, University of Copenhagen*. (Master thesis in Danish)
13. Vredenburg, K., Mao, J., Smith, P. W. and Carey, T. A Survey of User-Centered Design Practice. *Proc. CHI 2002*, 471 - 478

Remote Usability Analysis of MultiModal Information Regarding User Behaviour

Fabio Paternò, Angela Piruzza and Carmen Santoro
ISTI-CNR
Via G. Moruzzi, 1 Pisa (ITALY)
Fabio.paterno@isti.cnr.it
39 050 315 3066

ABSTRACT
In this paper we describe MultiModal WebRemUsine, a tool for remote usability evaluation of web sites that considers data coming from log files, videos recorded during user tests, and an eye-tracker. The tool performs an automatic evaluation of the usability of the considered web site by comparing such data (which describe the actual behaviour of the users) with that contained in the task model associated with the pages (which describes the expected behaviour of the user). The results of the analysis performed by the tool are provided to the evaluators in terms of task not completed, errors occurring during the performance of tasks, time for completing a task, etc. These results are provided along with information regarding the user behaviour during the task performance. Using such data, evaluators should be in a position to identify problematic parts of the website and make improvements, when necessary. An example of application of the proposed method is also shown in the paper.

Author Keywords
Remote usability evaluation, websites, multimodal data

ACM Classification Keywords
H5.m. Information interfaces and presentation (e.g., HCI): Miscellaneous.

INTRODUCTION

The dissemination of Web applications is enormous and still growing. The great penetration of Web sites raises a number of challenges for usability evaluators. In this paper we discuss what information can be provided by automatic tools able to process multimodal information on users gathered from different sources, so as to provide the most effective remote usability evaluation of websites. The collected information ranges from browser logs to videos to eye-tracking data, and the approach proposed tries to integrate such data in order to derive the most complete information for analysing, interpreting and evaluating the users while visiting a website, by taking into account the factors that might affect the performance of the users.

The proposed approach is supported by a tool – MultiModal WebRemUsine, which has been improved over the years in order to include and handle more and more information and provide additional features. In one of its first versions [4], the tool was just able to automatically analyse the information contained in Web browser logs and compare it with task models specifying the ideal behaviour of users interacting with the application and representing the actual Web site design. The goal was to identify where users interactions deviate from those envisioned by the system design and represented in the model. However, such information can be rather limited because when users visit a web page, their attention can be captured by different areas of the same page and this information cannot be derived just analysing log files, which are only able to track physical interactions of the user with the application (eg.: scrolling, clicking, etc.). Eye tracking is a technique able to allow for deriving the current area of interest of the user by following the user's gaze. Thus, it helps evaluators in discovering the navigation strategies of the users visiting the web site and analysing the impact of different areas of the page. This enables easy identification of possible problematic parts of the page.

However, there are situations in which even the eye-tracker data may result inadequate to provide sufficient information for effective evaluation. Indeed, a user may look at the same portion of the page for quite different reasons, and such reasons could not be discovered by just analysing eye-tracking information. For instance, users might delay in staring at a certain point of the page because they might not be aware of having found the requested information and still look for it or, alternatively, they are aware of having achieved the goal and thus are interested in further reading the information found. In both cases, a video-based analysis might provide useful information for interpreting the different impacts of the same page on users: for instance, it might highlight situations where, although the logged information and user's gaze might make evaluators conclude that the user has successfully completed the expected task, a puzzled expression of the user should force the evaluator to re-interpret the collected data and derive, more realistically, that the user has completed the task but might not realised it, which is still a signal of a usability problem in the page. So, this simple example shows how important is integrating all the data that is possible to capture on the user behaviour (and possibly also on the environmental/contextual conditions in which the user interaction takes place) in order to perform the most comprehensive evaluation.

The novelty of the proposed approach is the integration of data coming from various sources (eye-tracker, webcam, …) and the data coming from a tool able to know the current goal of the user and the related activities performed by the user in order to reach such goal. While such data have been already used in previous evaluation approaches in a separate manner, in our approach the added value is represented by integrating the two techniques. In this way, the data coming from the various sources allow for better understanding the information regarding the task the user is currently performing. Thus, the evaluator is provided with a more comprehensive picture of the actions performed by the user and consequently with more information in order to effectively interpret and evaluate the related user interface. Moreover, it is worth pointing out that, apart from the data provided by the eye-tracker, which still remains a rather expensive technology, the approach proposed has the remarkable advantage to allow evaluators to identify usability problems even if the analysis is performed remotely, which might contribute to keep at minimum the evaluation costs and allows the users to remain in their familiar environments during the evaluation, improving the trustworthiness of the evaluation itself.

In the paper, we first discuss related work, next we indicate the method underlying our environment and recall the main features of the first versions of WebRemUsine. Then, we present the new features of the new tool, MultiModal WebRemUsine, and report on some example application. Lastly, some conclusions along with indications for future work are provided.

RELATED WORK

Creating a Web site allows millions of potential users, who have diverse goals and knowledge levels, to access the information that it contains. While a Web site can easily be developed using one of the many tools available able to generate HTML from various types of specifications, obtaining usable Web sites is still difficult. Indeed, when users navigate through the Web they often encounter problems in finding the desired information or performing the desired task. With over 30 million Web sites in existence, Web sites have become the most prevalent and varied form of human-computer interface. At the same time, with so many Web pages being designed and maintained, there will never be a sufficient number of professionals to adequately address usability issues without automation [2] as a critical component of their approach. For these reasons, interest in automatic support for usability evaluation of Web sites is rapidly increasing [6][1].

With the advent of the Web and the refinement of instrumentation and monitoring tools, user interactions are being captured on a much larger scale than ever before. In order to obtain meaningful evaluation it is important that users interact with the application in their daily environment. Since it is impractical to have evaluators directly observe users' interactions, interest in remote evaluation has been increasing. In addition, some studies [7] have confirmed the validity of remote evaluation in the field of Web site usability. Some work [3] in this area has been oriented to using audio and video capture for qualitative analysis performed by evaluators on the result of usability testing. In our case we focus more on quantitative data (eg: log files and data coming from eye-tracking tools like length of the scanpath, duration of fixations, etc.) and provide the support for an intelligent analysis of such data so as to extract useful information for evaluation goals. The aim of this paper is to present a tool for performing remote usability evaluation of Web applications. We describe how it supports analysis of task performance and Web pages accesses by single users and groups of users.

THE ARCHITECTURE

In its first version, the core of the system architecture of our system was mainly composed of three modules (see Figure 1): the ConcurTaskTrees editor (publicly available at http://giove.cnuce.cnr.it/ctte.html) developed in our group; the logging tool that has been implemented by a combination of Javascript and applet Java to record user interactions; and WebRemUSINE, a Java tool able to perform an analysis of the files generated by the logging tool using the task model created with the CTTE tool.

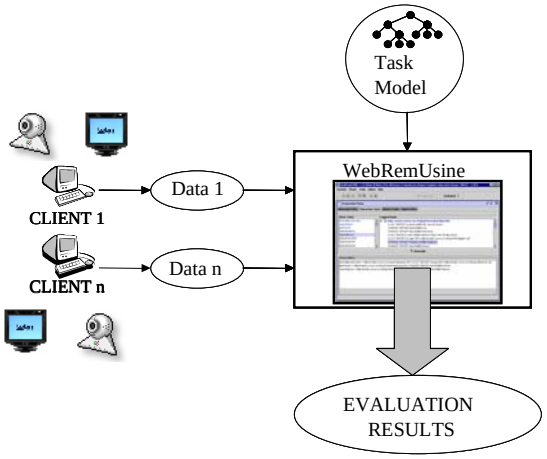


Figure 1: An Overview of the Architecture

The ConcurTaskTrees Environment (CTTE) is a tool for editing and analysing task models, which describe the activities to perform in order to reach user's goals. CTTE supports the ConcurTaskTrees (CTT) notation [5] for specifying task models, by providing a graphical representation of the hierarchical logical structure of the task model. In particular, it is possible to specify a number of flexible temporal relations among tasks (concurrency, choice, enabling, disabling, suspend-resume, order-independence, optionality, …) and for each task it is possible to indicate the objects to be manipulated and a number of other attributes. The notation also allows designers to indicate how the performance of the task should be allocated (to the user, to the system, to their interaction) through different icons.

The logging tool stores various events detected by a browser, using Javascripts that are encapsulated in the HTML pages and executed by the browser. When the browser detects an event, it notifies the script for handling it. By exploiting this communication, the script can capture the events detected by the browser and add a temporal indication. Then, a Java applet stores the log files directly in the application server. Our browser logging tool overcomes the limitations of other approaches to logging: server logs have limited validity since various page accesses are hidden to them because of browser cache memory and they do not detect the local interactions with the user interface elements (check-boxes, type in fields, …) that also in the case of proxy-based approaches cannot be detected.

The WebRemUSINE analysis can detect usability problems such as tasks with long performance or tasks not performed according to the task model corresponding to the Web site design. The task model describes how activities can be performed according to the current design and implementation. Either the designer or the evaluator can develop it. Since ConcurTaskTrees has various temporal and logical operators, it is possible to describe all the various paths the user can follow to accomplish a task. Deviations from the expected paths can be detected in the logs and represent useful information for identifying any pages that create problems to the user. Moreover, the tool analysis provides information concerning both tasks (such as time performance, errors, …) and Web pages (such as download and visit times, …). These results allow the evaluator to analyse the usability of the Web site from both viewpoints, for example comparing the time to perform a task with that for loading the pages involved in such a performance. WebRemUSINE also identifies the sequences of tasks performed and pages visited and is able to identify patterns of use, to evaluate if the user has performed the correct sequence of tasks according to the current goal and to count the useless actions performed. In addition, it is also able to indicate what tasks have been completed, those started but not completed and those never tried. This information is also useful for Web pages: never accessed Web pages can indicate that either such pages are not interesting or that are difficult to reach. All these results can be provided for both a single user session and a group of sessions. The latter case is useful to understand if a certain problem occurs often or is limited to specific users in particular circumstances.

THE UNDERLYING METHOD

The method underlying the tool is composed of two main phases, the preparation and the evaluation.

Preparation Phase

The main goal of the preparation phase is to create an association between the basic tasks of the task model and the events that can be generated during a session with the Web site. This association allows the tool to use the semantic information contained in the task model to analyse the sequence of user interactions. On the one hand, in the task model, all the activities that are supposed to be performed in order to reach a goal are hierarchically described (for an example of task model see Figure 3). In particular, basic tasks are tasks that cannot be further decomposed, whereas high-level tasks are complex activities composed of sub-activities. On the other hand, the log files are composed of sets of events. If an event is not associated with any basic task, it means that either the task model is not sufficiently detailed, or the action is erroneous because the application design does not call for its occurrence. For example, when a user sends a form then two events are stored in the log: one associated with the selection of the Submit button and the other one with the actual transmission of the form. Thus, in the task model two basic tasks are required: one interaction task for the button selection and one system task for the form transmission, otherwise the model is incomplete.

In the logs, there are three types of events: user-generated events (such as click, change), page-generated events (associated with loading and sending of pages and forms) and events associated with the change in the target task by the user, which is explicitly indicated through selection from the list of supported tasks.

Tasks can belong to three different categories according to the allocation of their performance: user tasks are only internal cognitive activities that thus cannot be captured in system logs, interaction tasks are associated with user interactions (click, change, …) and system tasks are associated with the internal browser generated events. In addition, the high-level tasks in the model are those that can be selected as target tasks by the user. The different allocation of the various tasks is represented by using different icons in the task model (see Figure 3). In order to ease the development of the task models we also developed a reverse engineering tool able to create automatically the task model corresponding to a web site implemented in (X)HTML. This model can be further refined by the designer, if necessary.

Each event is associated with a single task whereas a task can be performed through different events. For example, the movement from one field to another one within a form can be performed by mouse, arrow key or Tab key. The one-to-many association between tasks and events is also useful to simplify the task model when large Web sites are considered so that we need only one task in the model to represent the performance of the same task on multiple Web pages.

Once the association between tasks and events has been carried out, it is possible to move on the next step of the approach, the proper evaluation, which will be detailed in the next section.

Evaluation Phase

In the evaluation phase the proper automatic analysis is performed, where WebRemUSINE examines the logged data with the support of the task model and provides a number of results concerning the performed tasks, errors, loading time. WebRemUSINE displays all results in various formats both textual and graphical. Such information generated is analysed by the evaluators to

identify usability problems and possible improvements in the interface design.

During the test phase all the user actions are automatically recorded, including those associated to goal achievement. The evaluation performed by WebRemUsine mainly consists in analysing such sequences of actions to determine whether the user has correctly performed the tasks complying with the temporal relationships defined in the task model or some errors occurred. In addition, the tool evaluates whether the user is able to reach the goals and if the actions performed are actually useful to reach the predefined goals, or a precondition error occurred, which means that the execution task order did not respect the relations defined in the system design model.

In addition to the detailed analysis of the sequence of tasks performed by the user, evaluators are provided with some results that provide an overall view of the entire session considered:

- The basic tasks that are performed correctly and how many times they have been performed correctly.
- The basic tasks that the user tried to perform when they were disabled, thus generating a precondition error, and the number of times the error occurred.
- The list of tasks never performed either because never tried or because of precondition errors.
- The patterns (sequences of repeated tasks) occurred during the session ad their frequency.

Such information allows the evaluator to identify what tasks are easily performed and what tasks create problems to the user. Moreover, revealing tasks that are never performed can be useful to identify parts of the application that are difficult to comprehend or reach. On the basis of such information the evaluator can decide to redesign the site in order to reduce the number and complexity of the activities to be performed.

From the log analysis the tool can generate various types of results:

- *Success*: the user has been able to perform a set of basic tasks required to accomplish the target task and thus achieve the goal.
- *Failure*: the users starts the performance of the target task but is not able to complete it;
- *Useless uncritical task*: the user performs a task that is not strictly useful to accomplish the target task but does not prevent its completion.
- *Deviation from the target task*: in a situation where the target task is enabled and the user performs a basic task whose effect is to disable it. This shows a problematic situation since the user is getting farther away from the main goal in addition to performing useless actions.
- *Inaccessible task*: when the user is never able to enable a certain target task.

A further type of information considered during the evaluation regards the task execution time, and the duration is calculated for both high level and basic tasks. The set of results regarding the execution time can provide information useful to understand what the most complicated tasks are or what tasks require, in any event, longer time to be performed. Longer execution time does not always imply complicated tasks, for example in some cases downloading time can be particularly high. WebRemusine provides also detailed information regarding downloading time so that evaluators can know its influence on the performance time. A further type of evaluation concerns the time associated with actions that generate errors. By analysing when errors occur, it is possible to determine if the user's performance improves over the session. If the errors concentrate during the initial part of the test and their number decreases over time, we can assume that the user interface is easy to learn.

Modifications of the Evaluation Phase for Enriching Information on Users: Data from Videos and Eye-tracker

In the previous sections we have provided an overview of the system, and, from the point of view of the client side, we have only considered the data coming from the log files. However, such information revealed insufficient to perform an effective evaluation because there might be a number of factors that might affect the performance of a user while interacting with an application and cannot be described just recording log files. Then, integration of such data with videos recorded during the sessions and data from eye-tracker was used to enrich the multimodal information that the system is able to gather on the user.

As for the videos, an association between task and video is automatically performed by the tool, thanks to the information regarding the starting/ending time of the different tasks. Indeed, as the whole session is recorded by a webcam, thanks to such times it is possible to split the video associated with the whole user session into different fragments related to the completion of the various tasks, together with the possibility of visualizing the related video with a suitable player in the tool, that can be activated/stopped by the evaluator. It is worth pointing out that, in order to do this it was necessary to modify the process of recording log files so as to save the information about such times. The data from videos are important in that they can provide more 'contextual' information during the performance of a task. Indeed, since the evaluation is remotely performed, the evaluator is not in a position to understand if any condition might have disturbed the performance of a task while the user visits the web site in his/her own environment. For instance, a high time (or, at least, a time higher than expected) for completing a task might not necessarily be brought about by a usability problem, but it may be caused by interruptions during the session occurring in the user's environment and due to some external factors. Another useful information that can be gained from videos are user comments, which sometimes can reveal that users are aware of having performed an error but cannot undo the related actions.

While videos provide more 'contextual' information regarding users, giving the means for correctly interpreting the user's actions, the eye-tracker provides technical measurements and traces of the routes that users follow while visiting a website. Indeed, the eye-tracker records the pauses and 'jumps' that normally occur when a user looks at a page (respectively called fixations and saccades), together with the so-called 'scanpaths', the traced routes of sequences of fixations and saccades, revealing the path that the user followed during the visit of the page. By superimposing the scanpath on the page it is possible to understand the strategies used by the user to visit the page. Indeed, the evaluator can understand where the user paused to look and, alternatively, the areas of the page that did not attract his/her attention. Moreover, having in mind the target task that the user should achieve, it might be relevant to analyse the areas around the links that should be followed in order to reach the expected goal, according to the task description specified in the task model. For instance, it might be relevant to analyse the time users spent looking at these area (duration of fixations), as well as the number of accesses to such areas (if any), which is given by the number of fixations. By examining such variables it is possible to understand if users found difficulties in exploring the page: for instance, if the time duration is high, it might be a sign of user's difficulty in elaborating the information; also the number of fixations might be a sign of problems because the occurrence of some fixations on certain areas might indicate that the users are confused and are not able to find the information that they are looking for. Moreover, also the scanpath might give useful information for the evaluation: for instance, a long scanpath might indicate that the structure underlying the page is rather complicated, while, vice versa, a very short scan path indicates a rather poor structure of the page.

In addition, it is worth pointing out that, in order to manage the information associated with the eye-tracker, it was needed to modify also the logging tool and, in particular, handling scrolling events. Indeed, as soon as a scroll event is recorded, also the extent of the shifting is recorded with respect to the top and bottom corner of the page, so as to reconstruct the actual area that the user was currently looking at. Then, as we considered different sources of data for the evaluation and in order to provide the most flexibility possible, different evaluation options were considered and included in the tool:

i)*Browser input*, showing information derived from analysing log files;

ii)*Browser+WebCam* log files and users' video recorded during the tests;

iii)*Browser+EyeTracker* input, with log files information and eye-tracker data;

iv)*Browser+WebCam+EyeTracker* with log files information, videos, and eye-tracker data.

AN EXAMPLE

In this section we show an example of application of the proposed evaluation method. The web site we considered (http://www.pisaonline.it) provides information about Pisa (in Fig. 2 the homepage is shown). The website is divided into four main sections: "Pisa da Visitare" (Visiting Pisa), "Pisa da Vivere" (Living in Pisa), Pisa da Studiare" (Studying in Pisa) e "Pisa Aziende" (Companies in Pisa). For sake of brevity in Figure 3 only the task model related to the "Pisa da Visitare" section is visualised.

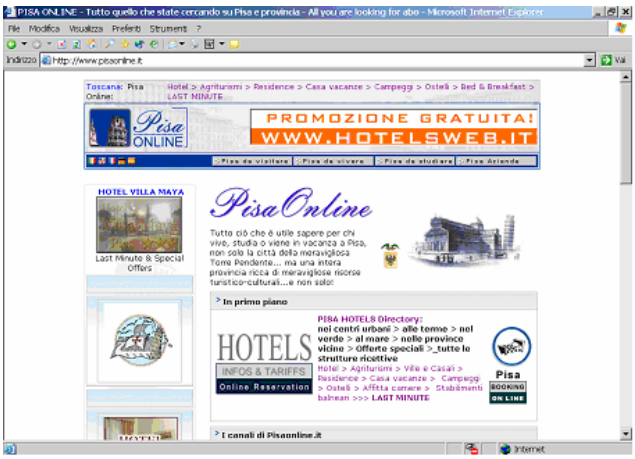


Figure 2: The homepage of the site evaluated

In the model the activities the user should perform to get information about places to visit in Pisa are described. For the evaluation several target tasks were identified, examples are "Trova Info su Tartufo Bianco" (Find information about the white truffle), "Trova gradazione alcoolica del Chianti", (Find alcoholic content of Chianti), etc..

Figure 3: The task model of the "Pisa da Visitare" section

We involved in the test participants aging between 21 and 36. In Fig. 4 the scanpath length of the homepage is visualised: as you can note, it is rather long (16554 pixels).

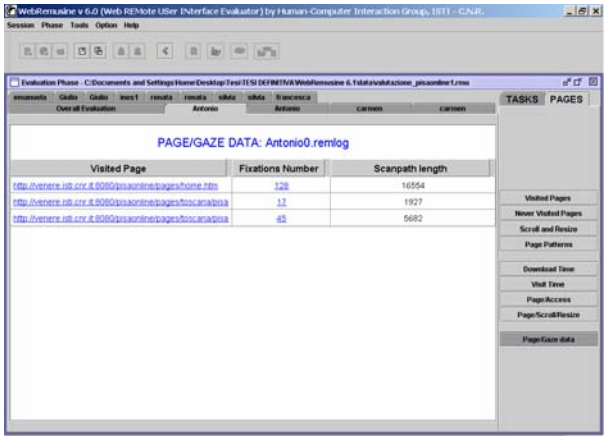


Fig. 4: The tool window associated with Page/Gaze data

Indeed, looking at the image of the scanpath superimposed on the homepage for the task "Find restaurant" (it is visualised in Fig. 5) it is possible to see that the gaze of the user moves from one point to another indicating a possible confusion of the user who seems not able to find the right link for carrying out the task.

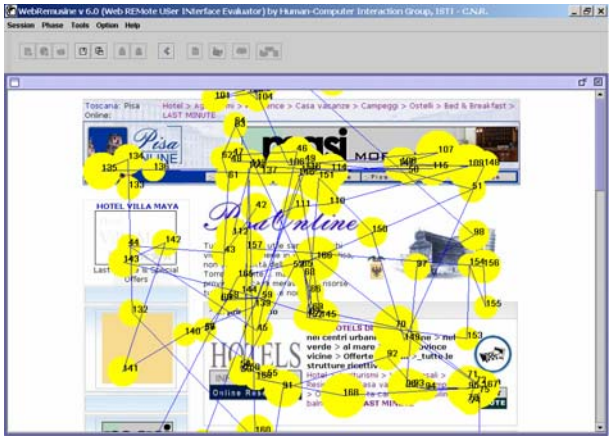


Fig. 5: Scanpath of "Find Restaurant"

Another user (who made the greatest number of errors during the navigation) selected as target task "Find information on the white truffle", which was associated with the abstract task "Access to Ulisse" ("Ulisse" is the name of an airline magazine providing general tourist information to their passengers). During the test such user performed a number of errors of precondition, because he carried out a number of tasks that were not necessary according to the designer's task model for achieving the task goal.

While trying to perform the task "Find Information on the White Truffle" the first of such errors was carried out because the user was observed pausing a lot while looking at the area of the page dedicated to the restaurants (a fixation with a relevant duration was registered), whereas s/he did not look at all to the right link (the one related to "Pisa da visitare") which, according to the designer's task model, represents the right route for completing the selected task ("Find Information on the White Truffle"). From this it might be derived that the user was wrong at interpreting the name "White Truffle" and interpreted it as a name of a restaurant.

In addition, after spending long time before selecting "Pisa da Visitare" link it is possible to note from the associated scanpath visualised in Figure 6 that the user found it difficult also to identify the right link among those associated with the "Access to Ulisse" target task. Indeed, in order to access to the section devoted to Ulisse, in the page visulaised in Fig. 6 there is a textual link (with label "Ulisse"), another textual link with a different label ("Alitalia Ulisse"), and also an icon with an image associated to Ulisse, and the information available through the last two links is different from that that can be accessed through the first link.

Fig. 6: The ambiguity of links related to the section dedicated to "Ulisse"

Moreover, the experiment highlighted that the majority of users did not select the image link associated with the homepage of the PisaOnline web site (visualised in the top left part of the homepage, see Figure 6) and associated with the related task "Seleziona Home" (Select home). It was rather surprising due to the relevancy of this page within the entire site. The occurrence of such behaviour in almost all users can be interpreted with the fact that the link is rather unclear, and this intuition is reinforced by the image related to the scanpath of users on the page (eg: see figure 6), highlighting that almost all users did not pause on looking at the concerned link. Indeed, it might be explained with the fact that the imagelink might have been confused with a bare decorative image, (especially because it appears on the top part of the page).

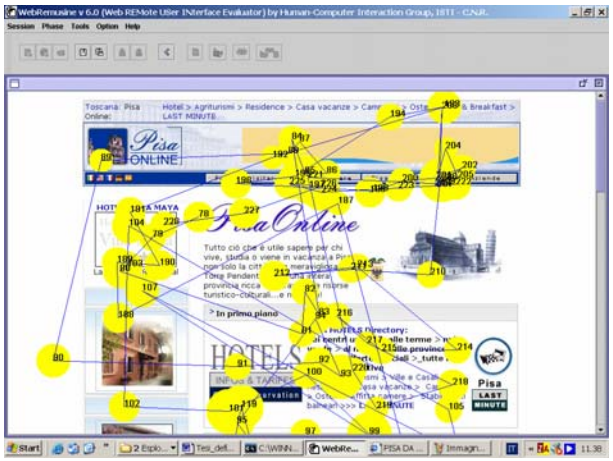


Fig. 7: Scanpath of "Find Alchoolic Content of Chianti"

In addition, users who selected "Find alchoolic content of Chianti wine" ("Trova gradazione alcolica del Chianti", see Fig. 7) as target task, which is related to the high-level task "Scegli vino" (Select wine) tended to find such information within the section called "Companies in Pisa", rather than, more correctly, within the "Living in Pisa" section, where the right link actually is. Figure 7 shows that there are many fixations (202,204,205,222,etc.) on the link associated with "Pisa Aziende" ("Companies in Pisa"), which highlighted that the logic followed by users in finding such information while exploring the page was different from that followed by designers.

In another experiment we analysed a different site regarding a publishing house and mainly focused on data recorded by videos. In Figure 8 you can see the evaluation of task/time performed by the tool, regarding a user who explicitly declared at the end of the task that she was wrong at completing the task.

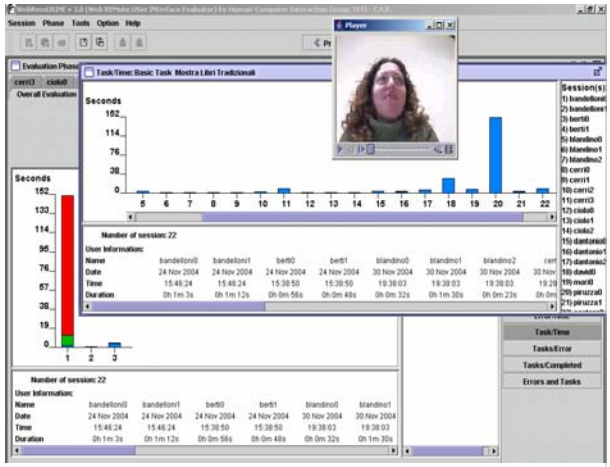


Fig. 8: Task/Time Information with Video of the user

The data from the videos were useful to detect some usability problems. For instance, by examining the tasks that were wrongly performed, it was possible to have further details on the facial expressions of the users, who sometimes seemed to be confident of their choices, while other times seemed to be quite confused and doubtful, and this information is important when evaluating the user behaviour. Particularly useful information was gained from videos as far as the execution time, which sometimes seemed to be higher than expected: the analysis of the video revealed that users pause at looking the page, then they happen to comment on it, so this is important to understand that it is not completely effective measuring the degree of difficulty of a task by just calculating the time spent in performing it, because users often are distracted/attracted by portions of the page that are not relevant for carrying out the concerned task, only by curiosity.

CONCLUSIONS

The advances in technology is more and more allowing evaluators to afford sophisticated hardware and software able to collect information about remote users interacting with web pages. In this paper, we propose a method, and the associated tool, for remote evaluation of websites that, through a combination of different sources of data coming from the client side (log files, videos and eye-tracker data) allows the evaluator to get detailed information about the behaviour of the users. More specifically, the evaluator can be informed on the page the user was visiting to perform a certain task, the sequence of actions carried out by the user on the page, together with detailed information of the route traced by the user gazing at the page and background videos on the user during the session.

Such composite information is the input of an automatic tool that has been developed in our group and has shown to be effective in providing evaluators with means for discovering possible problematic areas of the website.

Future work will be dedicated to extending the data regarding the user behaviour and state, including the emotional state, in order to have a more complete analysis of what happens during task accomplishment and better identify the potential usability issues

ACKNOWLEDGMENTS

We thank the SIMILAR European Network of Excellence for partly supporting this work.

REFERENCES

1. Card, S., Pirolli, P., Van der Wege, M., Morrison, J., Reeder, R., Schraedley, P., Boshart, J., (2001) Information Scent as a Driver of Web Behavior Graphs: Results of a Protocol Analysis Method for Web Usability, Proceedings ACM CHI 2001, pp.498-504.
2. Ivory M. Y., Hearst M. A., (2001) The state of the art in automating usability evaluation of user interfaces. ACM Computing Surveys, 33(4), pp. 470-516, December 2001.
3. Lister M., (2003) Streaming Format Software for Usability Testing, Proceedings ACM CHI 2003, Extended Abstracts, pp.632-633.

4. L.Paganelli, F.Paternò, Tools for Remote Usability Evaluation of Web Applications through Browser Logs and Task Models, Behavior Research Methods, Instruments, and Computers, The Psychonomic Society Publications, 2003, 35 (3), pp.369-378, August 2003.
5. Paternò, F., (1999) Model-based design and evaluation of interactive applications, Springer Verlag, 1999. ISBN 1-85233-155-0.
6. Scholtz, J., Laskowski, S., Downey L., (1998) Developing usability tools and techniques for designing and testing web sites. Proceedings HFWeb'98 (Basking Ridge, NJ, June 1998). http://www.research.att.com/conf/hfweb/ proceedings/scholtz/index.html
7. Tullis, T, Fleischman, S., McNulty, M, Cianchette, C. and Bergel, M., (2002). An Empirical Comparison of Lab and Remote Usability Testing of Web Sites. Usability Professionals Conference, Pennsylvania, 2002.

Statechart Metrics for Usability Evaluation

My Appelgren
University of Iceland
Hjardarhaga 2-6
107 Reykjavík, Iceland

Ebba Thora Hvannberg
University of Iceland
Hjardarhaga 2-6
107 Reykjavík, Iceland
ebba@hi.is

ABSTRACT
This paper examines the efficiency of using a review of UML statechart diagrams as an evaluation method for usability with respect to heuristic evaluation. By analysing and comparing test results from heuristic evaluations with the corresponding statechart diagram, we have defined metrics that indicate complexity. Results imply that statechart reviews function as a more thorough method for detecting problems concerning the structure of behaviours, whereas the user-visible design-aspects are its limitations.

Author Keywords
Heuristics Evaluation, UML

ACM Classification Keywords
H.5.2 [Information Interfaces and Presentation]:User Interfaces -- Evaluation/methodology

D.2.8 [Software Engineering]: Metrics, D.2.2 [Software Engineering]:Design Tools and Techniques -- State Diagrams

INTRODUCTION

Various studies have shown that usability tests involving users are effective when it comes to finding real usability problems. Still heuristic evaluation remains a popular method for evaluating software. It is a discount inspection method since it allows analysts to limit the scope of what they need to consider [3]. Hence, it takes less effort in terms of human resources and time. Yet, the predictability of heuristic evaluation with respect to user testing has been low [3, 10].

Previous studies have suggested different forms of formal specification for describing human-computer interaction with various types of models, such as cognitive, motor, perception, task, state, object/class and interaction models. In particular, formalisms have been suggested to describe the different states of a user interfaces together with the events that invoke actions and that stimulate transitions between states [8, 9]. For an overview of dialogue models, see Dix et al. [5]. The usage of models has been proposed as a part of interactive systems analysis, both task models [13, 16] and state models [11, 15]. With the increasing popularity and number of web sites, metrics have also been defined for analysis of web architectures represented as graphs[18]. The analysis has been performed with the aim of discovering anomalies, proving properties such as deadlocks, the ability to reach certain contexts in the user interface, robustness, consistency etc. or to quantify formal models. The analysis has been done manually with the aid of models, e.g. in a walkthrough, or with automatic tools, such as model checkers.

In this paper we like to examine whether predictability of a review of statecharts is higher than, or can complement, heuristic evaluation. To address this problem, we initially compare problems found in heuristics evaluation with the problems discovered during statechart review and ask:

- Are there classes of problems that statechart diagrams intrinsically cannot show?
- What are the classes of problems that statechart diagrams can show easily?
- Are there problems found through review of statechart diagrams that do not show up in the problem list of the corresponding heuristic evaluation?
- Can problems found through heuristic evaluation be validated in a statechart review?

We point out that there are certainly problems that neither heuristics evaluation nor statecharts can discover and a comparison with usability problems reported by users may reveal such a gap. Furthermore, we want to examine whether it is possible to develop a specific set of heuristics, metrics and/or formulas, based on the statecharts, that predicts the complexity of a user interface.

- Could heuristics, metrics and/or formulas, based on statechart diagrams, be used as a foundation/indication to predict usability problems?.

STATECHARTS

Finite state machines have traditionally been described using state transition diagrams. Statecharts have been proposed as improvements to state transition diagrams, among others to accommodate abstraction and modularity. Statecharts appear in different modelling languages such as the Unified Modelling Language (UML).

Horrocks [9] describes the event-state-action paradigm as events generated by users determining, together with the current state, what action(s) to perform and to what next state to go to. Thus, a user event will trigger the software

to move from one finite set of events to another. The user interface defines a set of possible events that a user can supply. Thus, a state defines the context in which an event takes place.

Statecharts can be drawn with different levels of abstraction, revealing more or less of the functionality and behaviour of the system. As with any other language constructs that can express different levels of abstractions, such as modules and classes, statecharts can express dynamic behaviour at different abstraction levels. When using the metrics presented below it is important to remember to use the same level of abstraction for all diagrams to be compared.

At the onset of a user interface design, one may choose an abstract description of the user interface, e.g. not detailing how a certain state is implemented. We can start to sketch statecharts as soon as we have determined concepts or objects of the user interface and want to model how they respond to events that are stimulated by users or other objects or even other systems. When we need to model the dynamic behaviour and understand how objects interacts or objects or the user interacts with objects, it is beneficial to sketch a state chart. In subsequent refinements, the behaviour of the states is modelled with more detail. A statechart is typically drawn in the dialogue phase, i.e. when the conversation between human and computer is designed. In general, the highest level of detail is always recommended for the most complete description. Note, that for the purpose of the study described in this paper, we decided to reverse engineer the statecharts, because we had already designed and implemented the user interfaces.

HEURISTICS EVALUATION

This paper compares data on usability problems discovered in a heuristics evaluation to data on usability problems discovered during a state chart review. The application that was evaluated is a learning resource broker, Educanext (www.educanext.org). 20 novice participants that were university students with background in computer science did heuristics evaluation. They received training on heuristics evaluation before the evaluation. 10 of participants applied heuristics evaluation using Nielsen's 10 heuristics [14] but 10 of them applied Gerhardt-Powals [7] cognitive principles. A more detailed account of the results of the heuristics evaluation is forthcoming.

LIMITATIONS AND BENEFITS OF STATECHARTS

The following two paragraphs describe the limitations and benefits of the statecharts' ability of finding usability problems, as found through our rigorous analysis of the statecharts drawn for the case study. The analysis primarily looked for returning patterns in the statechart diagrams where usability problems were found according to the heuristics evaluation.

Statechart diagrams provide a method for presenting user interface design in a methodological way. Thus, they are unable to detect problems concerning any of the following. (i) The user-visible aspects of the interface, such as selection of colours, fonts, placement on screen etc. (ii) Flaws in the underlying algorithms, as their design is not displayed in the diagrams. (iii) Missing features, if the functionality of a task is not directly dependent on the feature. From this we can conclude that the usability problems found with the help of statecharts are limited to issues concerning the structure of behaviours.

As for its benefits, we found that statecharts could identify problematic sections by concentrating on the following features. (i) The depth of the diagrams, as it implies the degree of software interaction; where deeper diagrams indicate increased complexity. (ii) Concurrent states, as they allow several states to be executed simultaneously and thereby increase the complexity of the user interface somewhat. (iii) Dead states, i.e. states that cannot be accessed, or from which a final state cannot be reached, as they must not exist in a statechart diagram. If they do occur, this is a severe problem. Dead states confuse, delay and sometimes even prevent the user from completing their task. Thus, this must be considered as a big drawback on the usability. (iv) Clustered states, which, opposite to the above-mentioned features, amplify the usability, as they give the user freedom to perform given tasks in any order. Directly clustered states are states immediately reachable when entering a specific context, thus they indicate ease of use. By looking at these features, we will also be able to identify potentially time consuming tasks.

METRICS AND REVIEW PROCEDURE

Based on the above findings of the limitations and benefits of statecharts, we have defined a set of metrics to be used in a statechart review. Table 1 presents the metrics and their indication, where items 1-5 are previously defined by Genero, M., et al. [6] for measuring complexity of statechart diagrams and items 6-11 are a contribution of this study. Genero et al. have used Briand et al.'s [2] property-based framework for theoretical validation and Poels and Dedene's [17], DISTANCE, as a measurement theory-based framework. Genero et al. have validated their metrics by empirically comparing them to understandability ratings.

The additional metrics proposed in this paper are influenced by statechart constructs, such as clustered states, depth of nested states and dead states that are undesirable. As for its benefits, we found that statecharts could identify problematic sections specifically by concentrating on the following features. (i) The depth of the diagrams, as it implies the degree of software interaction, where deeper diagrams indicate increased complexity. (ii) Guards, as they make it more difficult for users to understand how things are related. (iii) Actions, as they do not only bring you from one state to another, but also add a sometimes-unexpected event. (iv) Composite states, as they cluster states together, leaving the outgoing transitions dependent on a number or states and thereby increase the complexity of the user interface somewhat. (v) Action states, as they are difficult to maintain. During maintenance, transitions might need to be added, redirected and/or modified, which make action

states more difficult to use. This is because they should be given transitions only if the action included in the states always correlates to the incoming transition. If the action is placed on the transition instead, this problem will always be avoided. (vi) Dead states, i.e. states from which a final state cannot be reached, as they must not exist in a statechart diagram. If they do occur, this is a severe problem. Dead states confuse, delay and sometimes even prevent the user from completing their task. Thus, this must be considered as a big drawback on the usability. (vii) Concurrent states which, opposite to the above-mentioned features, amplifies the usability, as they give the user freedom to perform given tasks in any order. Directly concurrent parts are parts directly reachable when entering a specific context, thus they indicate ease of use. Further empirical research may suggest that other statechart constructs, such as history states, transient states, delays/time-outs, event priorities or parameterized states [9] may facilitate or hinder usability. This is a subject for further empirical experiments.

No.	**Metric Name**	**Metric Definition**	**Metric Indication**
1	NSS	The total number of simple states	Size metric
2	NCS	The total number of composite states	Complexity
3	NE	The total number of events	Size metric
4	NG	The total number of guard conditions	Complexity
5	NA	The total number of actions	Complexity
6	NCC	The total number of concurrent states	Complexity
7	NDS	The total number of dead states	Flaw in design
8	NDCP	The total number of direct concurrent parts	Ease of use
9	NAS	Total number of Action States	Complexity
10	LEP	Longest minimum Error Free Path	Time Consumption
11	D	Depth of the Diagram	Complexity

Table 1. Usability metrics for UML statechart diagrams

Each of the metrics can be examined individually and compared to see what influences or predicts the number of problems. This paper proposes three formulas to better abstract composite metrics. These are Complexity, Reachability and Time consumption. Before each formula, we try to explain the terms informally.

Complexity: A measure of the complexity of the structure of the statechart. An example of such a measure is McCabe's cyclomatic complexity [12] of a program component. Here, complexity refers to interaction complexity.

The complexity parameter is suggested, for enhanced identification of complex sections, where higher values reveal higher complexity. (see equation 1)

$$\text{Complexity} = (D + NCS + NCC + 2 \cdot NG + NA + 10 \cdot NDS + NAS - 2 \cdot NDCP) / 10 \quad (1)$$

The weights added in formula 1 justify the importance of certain features when assigning the complexity. One dead state causes more harm to the usability than one simple state. To balance this out, dead states have been weighted by ten. Similarly, the number of guards have been weighted by two as they have shown tendencies on influencing the usability more than then non-weighted metrics. The direct concurrent parts simplifies the usability to a further extent than for instance simple states complicate the same, hence the weight.

Additionally, by looking at the following features we will also be able to identify potentially time consuming tasks. (i) Simple states, as they represent the innermost states of the system. (ii) Events, as they represent the movement between the states. (iii) Path length, as long paths naturally imply higher time consumption. Longest error free path length implies the highest amount of direct event-transitions needed for completing a specific task in a context.

For the Time consumption construct, higher values indicate higher time consumption. When we present the metric Time consumption, we assume that the structure of the statechart can predict actual time spent in the user interface. It is similar to the assumption made in GOMS (Goals Operators, Methods and Selection) where it is assumed that the goal structure can yield measures of performance.

Time consumption: Assumes that structure of statechart can predict performance. The number of events, the length of paths to reach closure, and number of simple states.

(2) Time consumption = NSS + NE + LEP

For reachability, values closer to one suggest better reachability.

Reachability: Based on the assumption that if we can reach parts of the user interface concurrently (breadth), instead of sequentially (depth) this will make tasks more accessible and the user interface easier to use.

(3) Reachability = NDCP / NSS

Other metrics that are worth looking into in the context of state charts are cohesion and coupling [p. 423-441 4].

With a statechart review the primary focus is not on finding a specific number of usability problems the way you do in heuristics evaluation; instead, you will find a set of complex sections. To find the high complexity sections the following procedure is suggested:

1. Count the metrics as defined in Table 1.
2. Find the problematic sections with high complexity values according to Equation 1.
3. For each task in the problematic sections apply the following heuristics in your review:

   • Look for how feedback is resolved. Check if the method in use is the most appropriate for this purpose, and if there is any place with missing feedback. Hint: look for simple states with outgoing transitions to history states or self-transitions to history states.
   • Look for action states. Check if they could be resolved with actions on transitions instead. Action states could be very powerful, but are unfortunately also the cause of many problems, especially caused during system maintenance. Hint: Does the action apply to all incoming transitions under all circumstances? Is frequent maintenance likely to be needed?
   • Look for dead states. Hint: Look not only for states without outgoing transitions, but also for eventual guards that might block users from the possibility of leaving.
   • Look for long paths for completing a task. This could reveal a time consuming task.
   • Look for inconsistency. Is the same kind of task resolved differently in different contexts, or even within the same context? Look for similar frequently reoccurring tasks. Similarity makes it easier for the user.
   • Look for dependent concurrent parts. Dependency weakens the software design and makes the system more difficult to maintain. This solution should only be used as a last resort. Hint: Look for guards referring to states in concurrent parts.
   • Look for un-proportionally large concurrent parts. This could reveal a potentially time consuming and confusing task. **(2)**
   • Look for states that leave the main context during its paths while completing task.
   •Look for states that represent the same thing. Could any state be removed?
   •Look for states with multiple or unclear entries or exits.
   •Look for high numbers of guards on transitions; as users usually find increasingly difficulty with increased number of guards. **(3)**
   • Look for critical actions without guards that do not allow you to undo the action. Focus especially on the higher levels of abstraction.

4. List the problematic sections found for each tasks

VALIDATION - RESULTS

The data used in this study originates from heuristic evaluations of 20 participants of the learning resource broker, EducaNext (www.educanext.org). Based on the usability problems encountered, categories were created for both the contexts and the causes. A context here means a set of states of the user interface that implement one or more tasks. In the context category 'Handling Learning Resources', which contained one fifth of all the usability problems encountered, the problems that were directly connected to the tasks 'Provide a New Learning Resource' was extracted, analysed further and divided into 5 subcategories, or set of user interface states. The categories are: General (1), General (2), Technical, Educational and problems not specific to any context within the task 'Provide a New Learning Resource', General to LR. To give an idea of the contexts under consideration, General (1 and 2) are two data entry forms for meta-data about the learning resource, Technical is a data entry form on technical meta-data of a learning resource and finally Educational is meta-data concerning educational aspects of the learning resource. Problems of general nature, not classifiable into one of the other four contexts, are categorised into LR. Concurrently, the modeller drew nine statechart diagrams, reviewed them according to the procedure described in the previous section to find problematic areas, or high complexity sections, of the user interface and later compared the results to the heuristics evaluation problem list. The diagrams and further description of the results are available in [1]. The modeller that drew the statechart diagrams, by reverse engineering them from the user interface, did not have any part in the development of the user interface nor did she perform the heuristic evaluation.

Table 2 displays the outcome of the review and Table 3 displays the number of problems encountered with heuristics evaluation compared with the result of formula 1 for each context category.

	General (1)	General (2)	Technical	Educational	LR
NSS	38	109	32	24	251
NCS	4	9	3	1	30
NE	45	143	43	29	351
NG	7	28	3	0	64
NA	23	31	22	12	107
NDS	0	0	0	0	0
NCC	2	6	2	1	16
D	4	3	3	2	5
NDCP	13	16	8	12	13
NAS	0	0	0	0	0
LEP	3	8	2	1	43
Formula 1	2.6	7.5	2.1	-0.8	28.7
Formula 2	86	259	77	54	645
Formula 3	0.34	0.15	0.25	0.50	0.05

Table 2. Metrics result over five context categories

	General (1)	General (2)	Technical	Educational	LR
HE Problems	1	6	1	0	26.5
Formula 1	2.6	7.5	2.1	-0.8	28.7

Table 3. Weighted problems encountered with heuristic evaluation, user tests and formula result

Figure 1 shows the confirmed problems in the state chart review and the problems found with heuristics evaluation only. We do not report problems found with state chart review only, since we deem that assessment unreliable because an independent evaluator did not perform it. This will be the subject of further research.

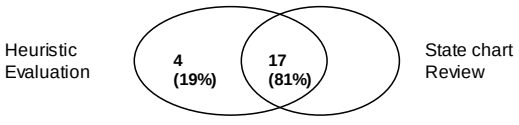


Figure 1 Problem coverage; heuristics evaluation and statechart review

Further examination of the severity of the problems, as assed by the evaluator, reveals that state chart review is able to uncover all severe problems, 67% of the moderate ones and finally 71% of the minor ones.

CONCLUSIONS AND FUTURE WORK

Below, we summarise answers to the questions posed in the introduction and conclude the paper with future work.

Are there classes of problems that statechart diagrams intrinsically cannot show? Yes. Analysis of the statechart diagrams proved that problems concerned with aesthetics or textual content could never be detected with a statechart review, nor could bad solutions and/or implementation of algorithms. Some of these problems, e.g. bad algorithms for search, cannot even be discovered by heuristics evaluation.

What are the classes of problems that statechart diagrams can show easily? Because of its composition, statecharts do provide a very efficient way of detecting usability problems concerning the structure of behaviours. Error identification and recognition of complex and time consuming tasks are all included in this problem range.

Are there problems found through review of statechart diagrams that do not show up in the problem list of the corresponding heuristic evaluation? Other than the fact that a statechart review might result in a more complete list of problematic sections, because of its absolute coverage, this study did not find any specific type of problems found with this method that heuristics evaluation cannot find.

Can problems found through heuristic evaluation be validated in a statechart review? Yes and no. The intersection between the problem lists of the two methods reveal that a total of 81 per cent of the problems found by heuristic evaluation could be validated by a statechart review. Interestingly all of the 10 severe problems were included in this share. As for the remaining 19 per cent, the statechart review is unable to validate.

Could heuristics, metrics and/or formulas, based on statechart diagrams, be used as a foundation/indication to predict usability problems? As far as this study concerns, results imply that the metrics developed, together with the formulas, give a good indication of where to find problems. The results from formula 1, representing the complexity of the user interface, projects very well with the heuristic evaluation problem list, suggesting it would function as a good indication on where to find complex areas. However, since the developed statechart heuristics have not been tested nothing more specific can be said about the problem identification efficiency for the statechart review.

The development of the metrics proposed in this paper for statecharts are still in progress and future work is therefore needed. The reliability of the statechart construction needs to be evaluated to see if two dialogue designers will produce similarly complex statecharts for the same user interface. It may be important to investigate this if we choose to evaluate the user interface when it has been built and reverse engineer the statecharts, but if the statecharts are used to guide the dialogue design from the onset, two designers cannot be expected to produce similar statecharts. The metrics themselves are counts and are therefore considered reliable. Since the statechart review heuristics are applied manually, they are subject to evaluators' thoroughness and knowledge. A separate experiment is needed to see how reliably they can be applied. Further heuristics evaluation, including severity assessment, was performed by novices and may therefore limit the validity of the study.

Continued research on the validity [19] of the metrics and the formula is needed. Internal validity, or cause-effect relationship, has to be addressed to examine whether there is a relationship between the metrics and other quality factors. Only one task, albeit a very large and problematic one, on the EducaNext learning resource broker has been examined. External validity, or generalizability, will be studied during further empirical research on other parts of EducaNext or different applications and more dialogue designers. Preliminary statistical measures have been calculated to assess the statistical conclusion validity but since this experiment only contained 9 diagrams, we will report the results once we have drawn additional diagrams and have more data.

The same kind of comparison as presented in this paper is intended for the usability problems reported during evaluation with user participation.

ACKNOWLEDGEMENT

We are grateful to the second author's co-authors of the study on heuristics evaluation. We are thankful to the reviewers for useful comments. This work has in part been sponsored by the Research Fund of RANNIS

REFERENCES

1. Appelgren, M. A Statechart Review for Predicting and Comparing Usability *Faculty of Engineering*, University of Iceland, Reykjavík, 2005, 86.
2. Briand, L.C., Sandro Morasca and Victor R. Basili Property-based software measurements. *IEEE Transactions on Software Engineering, 22* (1). 68-86.
3. Cockton, G. and Woolrych, A. Sale must end: should discount methods be cleared off HCI's shelves? *Interactions*, 2002, 13-18.
4. Constantine, L.L. and Lockwood, L.A.D. *Software for Use*. Addison Wesley, 1999.
5. Dix, A., Finlay, J., Abowd, G.D. and Beale, R. *Human-Computer Interaction*. Pearson, 2004.
6. Genero, M., Miranda, D. and Piattini, M. Defining Metrics for UML Statechart Diagrams in a Methodological Way. in *Conceptual Modeling for Novel Application Domains*, Springer Verlag, 2003, 118-128.
7. Gerhardt-Powals, J. Cognitive engineering principles for enhancing human-computer performance. *International Journal of Human-Computer Interaction, 8* (2). 189-211.
8. Harel, D. Statecharts: A Visual Formalism for Complex Systems. *Science of COmputing Programming, 8* (3). 231-274.
9. Horrocks, I. *Constructing the User Interface with Statecharts*. Addison-Wesley, Boston, 1999.
10. Law, E.L.-C. and Hvannberg, E.T., Analysis of strategies for improving and estimating the effectiveness of heuristic evaluation. in *Proceedings of the third Nordic conference on Human-computer interaction*, (Tampere, Finland, 2004), ACM Press, 241-250.
11. Loer, K. and Harrison, M. Formal Interactive Systems Analysis and Usability Inspection Methods: Two Incompatible Worlds. in Palanque, P. and Paterno, F. eds. *Proceedings of the Interactive Systems, Design, Specification and Verification. 7th International Workshop, DSV-IS 2000. Lecture Notes in Computer Science, vol. 1946*, Springer-Verlag, 2001, 169-190.
12. McCabe, T.J. A Complexity Measure. *IEEE Transactions on Software Engineering, 2* (4). 308-320.

13. Mori, G., Paterno, F. and Santoro, C. CTTE: Support for Developing and Analyzing Task Models for Interactive System Design. *IEEE Transactions on Software Engineering*, *28* (8). 797-813.
14. Nielsen, J. *Usability Engineering*. Academic Press, 1993.
15. Palanque, P. and Bastide, R. Synergistic modelling of tasks, users and systems using formal specification techniques. *Interacting with Computers*, *9*. 129-153.
16. Paternó, F. and Santoro, C. Preventing user errors by systematic analysis of deviations from the system task model. *International Journal of Human-Computer Studies*, *56* (2). 225-245.
17. Poels, G. and Dedene, G. Distance-based software measurement: necessary and sufficient properties for software measures. *Information and Software Technology*, *42*. 35–46.
18. Ricca, F. and Tonella, P., Analysis and testing of Web applications. in *Software Engineering, 2001. ICSE 2001. Proceedings of the 23rd International Conference on*, (Toronto, Ontario, Canada, 2001), IEEE, 25-34.
19. Straub, D.W. Validating Instruments in MIS Research. *MIS Quarterly*, *13* (2). 147-169.

Personality Responses During Human-Computer Interaction

Horia D. Pitariu
Babeş-Bolyai University, Department of Psychology
Cluj-Napoca, Romania

INTRODUCTION

Human-computer interaction (HCI) is defined by Card, Moran & Newell (1983, p. 4) as a process in which "the user and computer engage in a communicative dialogue whose purpose is the accomplishment of some task". This process is influenced both by the computer system design and by individual characteristics of the user. HCI has numerous technical, psychological and physiological implications. So we look at HCI as an interdisciplinary practice (Preece, Rogers, Sharp, Benyon, Holland, Carey, 1994).

In general, the influence of user characteristics or individual differences upon the way a user interacts with a computer system is an area in which research has been inconclusive, and in which little consensus has been reached (Brazier, 1991; Opperman, 1990; Van der Veer, 1990; 1989). Computer systems should be designed to enhance user performances but, unfortunately, the designer has in view only a limited number of individual characteristics, most of which refer to a prototypical user. The user's characteristics influence the communication within the Human-Computer (HC) system. They act through integrating themselves as a moderator variable in the performance. The aim of the present research is to uncover some personality components which can influence the user's mental model and the communication process between naive users and the computer system /a particular software. The main applications of this findings are in adaptive HCI design, computer help design, naive user's instructional strategies design etc.

METHOD

Participants. The participants were psychology freshmen who were enrolled in a basic statistics course and in an application-oriented course in data processing. The advantage of using psychology students as naive users was that most have had no previous experience with statistical software packages , so that experience will not be a confounding variable.

Data for this study was collected from a sample of 117 (17 male and 100 female) students. Their age ranged from 18 to 31, with a mean of 20.5 and a standard deviation of 2.61. Their experience with computers was minimal (this experience includes only a one-semester course, so they knew something of DOS and had some SPSS information).

Measurement.

Tests and questionnaires

The Eysenck Personality Questionnaire (EPQ) (Eysenck, Băban, Derevenco & Pitariu, 1989). The EPQ is one of the most frequently used personality questionnaires. In the EPQ four personality factors are distinguished, namely, psychoticism (P), extraversion (E), neuroticism (N), and social desirability or Lie Scale (L). Test norms for the EPQ have been developed in Romania. The Romanian EPQ version comprises 79 items.

The State-Trait Anxiety Inventory (STAI) (Spielberger, Gorsuch & Lushene, 1970; Kahana-Moscu & Vrânceanu, 1983). This inventory measures two distinct anxiety concepts: state anxiety (STAI-S) and trait anxiety (STAI-T).

The State-Trait Anger Expression Inventory (STAXI) (Spielberger, 1988). The STAXI provides measures of the experience and expression of anger. It is conceived as having two major components: state and trait. The STAXI consists of 44 items (each of them being scored on a 4-point scale) organized in six scales and two subscales:

State Anger (S-Anger): A 10-item scale which measures the intensity of angry feelings at a particular time.

Trait Anger (T-Anger): A 10-item scale which measures individual differences in the disposition to experience anger.

Angry Temperament (T-Anger/T): A 4-item subscale which measures a general propensity to experience and express anger without specific provocation.

Angry Reaction (T-Anger/R): A 4-item subscale which measures individual differences in the disposition to express anger when criticized or treated unfairly by other individuals.

Anger-In (AX/In): An 8-item anger expression scale which measures the frequency with which anger is held in or suppressed.

Anger-Out (AX/Out): An 8-item anger expression scale which measures how often an individual

expresses anger toward other people or objects in the environment.
Anger Control (AX/Con): An 8-item scale which measures the frequency with which an individual attempts to control the expression of anger.
Anger Expression (AX/EX): A research scale based on the responses to the 24 items of the AX/In, AX/Out and AX/Con scales which provides a general index of the frequency with which anger is expressed, regardless of the direction of expression.
The Computer Anxiety Rating Scale (CARS) (Heinessen, Glass, Knight, 1987). The CARS is a 19-item self-report inventory. Each item is scored on a 5-point scale. The scores can range from 19 (low computer anxiety) to 95 (high computer anxiety). We administered the CARS two times, at the start and at the end of the semester, after the students finished the 10 hours' SPSS course.

Performance variables
The Basic Statistics Knowledge Test (BSKT). This is a knowledge test in basic statistics, administered after the first semester.
General Computer Knowledge (GCK). The subjects responded on a five-point scale to the following statement: "I know a great deal about computers". The statement was included in the CARS, so it generated two answers, at the beginning of the course and at its end.
The Test of Users' Representation (TUR). The purpose of this test was to investigate and describe the structure of the user's mental model; it is a test that encourages the user to externalize the representation(s) of his personal mental model. This technique is known as "teach-back procedure" (Pask & Scott, 1972; Van der Veer, 1990). The TUR is scored on 4 different aspects: **Style** (verbal description, visual-spatial image, use of icons, set of production rules, programs); **Levels of Representation** (task level, semantic level, syntax level, key-stroke level); **Completeness**; **Correctness** (a thorough description of the components being measured by the TUR is presented by Van der Veer (1990)). The TUR was adapted after Moran (1981). As Van der Veer (1990) suggested, we excluded from Moran's components the physical or spatial layout level and the (hardware) device level. Several levels may be present in any individual representation.
The TUR protocols were scored according to a special procedure which ensured an appropriate statistical treatment.
The scoring system was the following:

Style of representation: For every possible style of representation we reckoned a score equal to (number of items where the style in question has been used / number of items where the style in point could have been used and which have been answered by the subject) * 100, if the subject has answered at least one item that could be solved in that style. If the subject didn't answer any of the test items apt to be solved in that style, he got a score of -1.

Levels of representation: For every possible level of representation we reckoned a score equal to (number of items where the representation level in question has been used / number of items where that representation level could have been used and which have been answered by the subject) * 100. If the subject didn't answer any of the test items apt to be solved at that level, he got a score of -1.

Completeness index = completeness score / 22 * 100.
(The completeness score is the number of items answered by the subject, whether the answer is right or wrong. 22 is the total number of items.)

Correctness index = correctness score / completeness score * 100.
(The correctness score is the number of items correctly answered by the subject.)
of these instruments.

Procedures
The experiment consisted of three parts:
(1) Testing (abilities, attitudes, personality)
(2) Teaching basic statistics and the fundamentals of the SPSS
(3) Testing knowledge in basic statistics and in SPSS.
The TUR was included in the final knowledge examination as a separate task.

RESULTS
A number of data analyses were carried out.
1. The analysis of personality measures
Some researches have suggested that anxiety and negative attitudes toward computers may adversely affect the user's motivation and performance in computer work (Shneiderman, 1979). Therefore we used the STAI (scales STAI-S and STAI-T) as a measure of anxiety, and the STAXI (scales T-Anger/T, T-Anger/R, AX/In, AX/Out, and AX/Con) as a measure of angry behavior.
Table 1 presents the statistical summary of STAI and STAXI.
Table 1
Statistical values of STAI and STAXI scales

Scale	Total Group			Males			Females			p
	N	M	SD	N	m	SD	N	m	SD	
STAI										
STAI-S	117	39	10.23	17	40	11.68	100	39	10.07	-
STAI-T	117	44	8.61	17	39	8.20	100	45	8.48	.05
STAXI										
T-ANGER/T	117	8	2,49	17	8	2.77	100	8	2.47	-
T-ANGER/R	117	10	2.51	17	10	2.20	100	10	2.57	-
AX/IN	117	18	3.36	17	17	3.82	100	18	3.31	-
AX/OUT	117	17	2.99	17	17	4.24	100	16	2.74	-
AX/CON	117	14	4.99	17	17	4.03	100	14	5.09	-

The EPQ, too, has been used to investigate the subjects' personality. Table 2 presents the EPQ scores. The mean scores on the P, N, and L scales show significant sex differences.

Between the EPQ scales we found some monotonous functional relations. The calculations have been carried out on the total group.

Table 2

Statistical values of EPQ scores

Scale	Total group			Males			Females			p
	N	M	SD	N	m	SD	N	m	SD	
P	110	2	1.59	15	3	1.88	95	2	1.43	.001
E	110	12	3.76	15	12	4.06	95	12	3.70	-
N	110	12	2.77	15	10	2.68	95	12	2.58	.001
L	110	8	3.68	15	6	3.51	95	9	3.53	.01

L depends linear-decreasingly upon P (r=-.33; p<.001). The L scale attempts to measure a tendency on the part of some subjects to "fake good"; it measures also some stable personality factor which may possibly explain some degree of social naivete (Eysenck & Eysenck, 1975). In other words, the L scale measures dissimulation which in our case is opposed to psychotic deterioration.

On the other hand, N depends linear-decreasingly upon E (r=-.33; p<.001).

Our data confirm the experimental data obtained by the authors of the EPQ (Eysenck & Eysenck, 1975).

Between the EPQ scales, the STAI-T variable and the variables of the STAXI questionnaire (T-Anger/T, Ax/In and AX/Con) we found a few monotonous functional relationships.

Thus, the N scale depends liniar-increasingly on STAI-T (r=.42, p<.001) and T-Anger/T (r=.39, p<.001), and liniar- decreasingly on AX/Con (r=-.34, p<.001). In other words, a high score on the neuroticism scale is connected with a high score on anxiety and angry behavior as a trait. Those persons with lover scores on the N scale try to control their anger.

The E scale depends linear-dedecreasingly on AX/In (r=-.28, p<.05). Subjects whose angry feelings are held in or suppressed manifest themselves as persons with an introverted behavior.

2. Computer anxiety measures

The CARS was developed as a general tool for estimating the level of computer anxiety (Heinessen, Glass & Knight, 1987).

Table 3 presents the statistical summary of CARS.

Table 3

Statistical values of CARS – Total scores

Variable	Total group			Males			Females			p
	N	M	SD	N	M	SD	N	m	SD	
CARS-1	115	43	8.09	17	39	9.17	98	44	7.70	.05
CARS-2	115	45	7.82	16	41	8.01	95	46	7.56	.05

Examination of the CARS scores across the two administrations revealed that the means for the two sexes are significantly different (p<.05). For total CARS, Cronbach's alpha ranged between .78 to .83, which is a high reliability.

We notice that the computer anxiety score obtained at the second administration of the CARS depends linear-increasingly (r=.72, p<.001) on the total score from the first testing: those who at first obtained high scores, i.e. exhibited high anxiety toward the computer, kept scoring high at the second testing as well.

For male students, the CARS values do not change significantly (p>.05) from one testing to the other, but for female students the mean difference between the two scores is significantly different from zero (p<.001) (N=95, mean=2.15, s=5.99). For both male and female students, mean scores slightly increased between the first and second administration. This might be ascribed, in part, to the unexpected difficulties confronting the students in the course of the training program in general, and the SPSS in particular. At the same time, anxiety toward the computer can be influenced by a situational factor (the second CARS administration took place shortly before the final exam).

The initial anxiety toward the computer is influenced, to some extent, also by state anger (the CARS-1 score depends linear-increasingly on S-Anger; r=.33, p<.001).

At the same time, CARS-1 depends linear-increasingly (r=.23, p<.05) on T-Anger/R (those who react angrily to a treatment they view as unfair manifest more fear toward the computer).

Factor analysis of the items included in the CARS revealed the existence of five factors, regardless of whether we deal with CARS-1 or CARS-2 (to be sure, the item grouping presents some unimportant differences between the two testings). These factors account for 58.4% (CARS-1) and, respectively, 62.8% (CARS-2) of the variance. Our discussion concerns only CARS-2. The names of the factors are the following (we preserved the item numbering from the original questionnaire of Heinessen, Glass and Knight, 1987):

F1 - the subject's being convinced that he is unable to learn the use of the computer (items 2, 3, 4, 5, 9, 10, 12 17, 18);

F2 - lack of assurance and fear to generate irremediable errors in computerized data processing (items 1, 2, 13, 14, 15);
F3 - feeling of inferiority with respect to the computer (items 8, 11);
F4 - the subject's belief that he cannot develop computer-operating skills solely by practice (items 6, 7);
F5 - failure to acknowledge the importance of computers in the work process (item 19).

We considered that the second administration of the CARS is more relevant, since it eliminates some of the initial distorting situational components, and because it is closer in time to the testing with the TUR (to be sure, the situational element is present, but it is related to the usual end-of-term examinations and, hence, it is a well-known variable).

We found several relationships between the CARS factors and the other variables under study. They are the following:

- F1 depends linear-decreasingly ($r=-.23$, $p<.05$) upon the GCK-2 score; that is to say, those who think they don't know much about computers consider they will not be able, either, to acquire the necessary knowledge. To put it another way, the factor in point expresses the subject's conviction that he cannot learn to work with the computer because he realizes that he is lacking in knowledge in this field.
- F2 depends linear-increasingly on the scales : N of the EPQ ($r=.31$, $p<.01$), STAI-T ($r=.37$, $p<.001$) and AX/In ($r=.28$, $p<.01$) and linear-decreasingly on GCK-2 ($r=-.45$, $p<.001$). The meaning of these relationships is that lack of assurance in operating the computer appears mainly in anxious persons, in those who are nervous and in those with scarce knowledge about computers.
- F3 depends nonlinear-increasingly upon the scale AX/Out. Complexes generated by confrontation with the computer are typical mainly of individuals who express anger toward other people or objects in their environment.

Except for factor F2, there are no significant ($p>.05$) sex differences in the mean factor scores. Factor F2 has the mean significantly higher ($p<.001$) for girls than for boys.

3. The Test of Users' Representation

The Test of Users' Representation (TUR) explores the user's mental model. This model is a result of learning activity or educational influences. From his mental model the user predicts the behavior of the system in interaction with his own behavior. The user's decisions are made with this model in mind (Van der Veer, 1990). In this sense, the user's performance is a variable dependent upon his mental model.

The TUR is, in fact, a knowledge test, but a peculiar one. The subjects are asked to perform an introspective activity and, subsequently, to report their thinking strategy. The work technique is a variant of the "verbal protocol", but much more operational and more accurate in scoring. Concretely, the subject is required to explain to a third person, by any means of communication he considers to be the most convincing, how he should operate with the computer and with the SPSS controls to carry out a few well-defined tasks (to write a program, to explain a program, to operate with a data base etc.). As a matter of fact, the subject's operational mental model is being tested. Quantification, that is, the rating system for the subjects' knowledge, has to be very accurate. It was, therefore, essential to find the most appropriate protocol-scoring system.

Teach-back protocols may be scored on four different aspects: **style of representation**, **level**, **completeness**, and **correctness**.

Table 4 presents the statistical summary of the TUR performances.

Table 4
Statistical values of mental model components

Variable	Total group			Males			Females		
	N	m	DS	N	M	DS	N	m	DS
Style of representation									
Verbal description	114	38	22.81	17	40	19.78	97	38	23.27
Visual-spatial image	90	40	44.99	15	51	48.48	75	38	43.93
Use of icons	104	44	39.07	16	53	41.34	88	43	38.42
Set of production rules	114	55	26.03	17	56	22.17	97	55	26.65
Programs	114	24	25.72	17	24	21.41	97	24	26.40
Levels of representation									
Task level	111	48	35.92	17	44	34.75	94	48	36.10
Semantic level	111	21	23.51	17	25	27.49	94	21	22.67
Syntax level	84	38	40.73	12	46	43.10	72	36	40.16
Key-stroke level	111	70	22.19	17	67	25.27	94	71	21.55
Completness index	115	49	18.85	17	48	19.63	98	49	18.71
Correctness index	115	61	18.43	17	60	18.19	98	61	18.46

With regard to the **Style of representation**, the mean scores of the two sexes do not differ significantly from each other ($p>.05$) for any of the styles. We notice high dispersion values, especially for the styles Visual-spatial image and Use of icons, and this for both sexes.

The lowest scores have been obtained for Programs.

As concerns **Levels of representation**, for none of the levels did we find significant sex differences ($p>.05$). The highest values have been obtained for Key-stroke level, and the lowest ones for the Semantic level. A possible explanation could be that the subjects' answers might reflect the training method which emphasized primarily the keys to be pressed in order to produce a particular command. Practically the subjects acquired a mechanical work algorithm rather than the logic of it.

The analysis of frequency distribution on **Completeness** and **Correctness** scores, in the total group, reveals the fact that the achievement test in

question was difficult. None of the subjects answered all the items, and nearly half of the subjects answered only 9, 10, 11 or 12 items out of 22. The mean score on **Correctness** is even very low (6.10), and more than half of the subjects are below this mean. The mean **Completeness** and **Correctness** scores show no significant ($p>.05$) differences between sexes.

4. Relationships of the mental model components with personality dimensions

From the study of dependences between the variables we can make a few inferences.

Style of representation

- **Verbal description** depends nonlinear-increasingly on the L scale of the EPQ. Subjects giving "fake good"- type responses are inclined to use more frequently **Verbal description**.
- The score on **Visual-spatial image** depends nonlinear-decreasingly upon the N scale of the EPQ and the F5 factor. Nervous subjects use more seldom **Visual-spatial images**. Those persons who don't recognize the value of the computer in the work process don't like to use **Visual-spatial images**.
- The **Use of icons** score depends linear-decreasingly upon the F2 factor ($r=-.22$, $p<.05$) and nonlinear-increasingly upon the F3 factor. Those who are uncertain in performing the computer operations and are afraid they could make errors seldom resort to **Use of icons.** Those persons affected by a computer complex prefer **Use of icons**.

  The **Use of icons** score is directly dependent on practice with computers. Those students reporting that they have used the computer only during the practice classes ($N=49$, $m=38.4$, $s=35.51$) resort less frequently to **Use of icons** in mentally representing the SPSS, than do those who have performed extra work with the computer ($N=15$, $m=61.1$, $s=41.57$) ($p<.05$).
- The **Set of production rules** score depends linear-increasingly ($r=.20$, $p<.05$) upon the F3 factor and nonlinear-increasingly upon the F4 factor. Subjects with an inferiority complex towards the computer prefer to use **Set of production rules**. Those subjects who believe that everybody can use a computer if he is motivated for it didn't like to use **Set of production rules**. They are likely to act in a somewhat superficial way in that they don't use the operating rules specific to the software, but instead prefer to learn to operate by the costlier "trial and error".
- The score on **Programs** depends nonlinear-decreasingly upon the F3 factor. The inferiority complex toward the computer is likely to take shape also in the fear of mentally representing by programs the tasks to be performed.

Levels of representation

- The **Task level** score depends linear-decreasingly on the F4 factor ($r=-.23$, $p<.05$). Those who believe that not everybody can learn, by regular practice, to work with the computer and the pertaining software prefer to reduce **Task level** representation. They have, then, only a superficial image about the SPSS (probably, this can be ascribed, in part, to the way of teaching the SPSS which emphasized primarily the acquisition of sequences of steps that represent operation commands at the menu level).

- The **Syntactic level** score depends linear-increasingly on the frequency of working with the SPSS ($r=.31$, $p<.05$). The more the frequency of operating with the informatics program increases, the more frequently we find operations at the **Syntactic level.**
- The **Key-stroke level** score depends nonlinear-increasingly upon the factors F1 and F4. Those convinced they cannot learn to use a computer and those believing that the SPSS cannot be learned by practice alone use more frequently in their representation system the **Key-stroke level.** They lower their work with SPSS to the level of mere key-pressing according to mechanically memorized operation sequences. .

Completeness index

The **Completeness index** depends nonlinear-increasingly on the scale STAI-T. In describing the SPSS program, anxious subjects make use of much more details than non-anxious ones.

Correctness index

The **Correctness index** depends nonlinear-decreasingly upon the N scale from the EPQ and linear-increasingly upon the BSKT ($r=.24$, $p<.05$). To put it concretely, the higher the mark on statistics, the higher the **Correctness index**. Persons who are worried and nervous have faulty representations about the software. The **GCK-2** score depends nonlinear-increasingly on the **Correctness index.** Persons having correct representations about the SPSS consider they know quite a lot about computers.

DISCUSSION AND CONCLUSIONS

Our study dealt with finding out some personality dimensions that may facilitate the interaction of the user's mental model with the user interface and the computing system in general. It is a well-known fact that individual characteristics are one of the causes for

success or failure in interactions that take place at the user interface. These characteristics differ in their resistance to change. The user's representations about the system differ under various aspects that can be measured separately. Thus, some users prefer to describe their interaction with the system at the key-stroke level, while others do it at the semantic level (through objects, attributes, actions etc.). Some users may describe the system correctly at the task and semantic level, but fail at the syntax and key-stroke level. All these findings are directly applicable to HCI design. The endeavors to design adaptive interfaces and optimum strategies for user training are a topical concern, and the present paper aims at making some contribution toward this goal.

The interaction between user and computer can be improved if psychological characteristics are taken into account in the design of the user interface and of the instructional strategies (Van der Veer, 1990). Some of these characteristics are more flexible, they can change over time; other ones form personality dimensions showing a high degree of stability. Thus the personality dimensions assessed by means of the EPQ are more stable, whereas some components of anxiety toward the computer can subside without any special or enduring effort.

Anxiety, no matter whether we speak about trait anxiety, that is, anxiety as a personality component, or state anxiety, i.e. situational/transitory anxiety (sometimes we mean by it some specific anxiety like computer anxiety), seems to be an important fact in HCI. The same can be said about angry behavior.

From a certain intensity level on, anxiety and angry behavior get a negative connotation, and under such circumstances the user's interaction with the computer is liable to failure.

From our study we were able to derive a few conclusions.

(1) What we called anxiety toward the computer turned out to have a multidimensional composition and to rest upon a certain personality structure of the user. Namely, it is influenced by stable anxious behavior (trait anxiety), as well as by neurotic behavior and by the frequency of the subject's suppressing his anger or directing it toward other people (AX/In and AX/Out)

However, we found also a balancing mechanism, and this is the influence of education (lack of knowledge about computers augments computer-phobic behavior, while its presence diminishes it).

(2) The components of the user's mental model in a software application are influenced by some dimensions of computer anxiety and by the experience acquired through operating the software in question.

A weighty influence upon the interaction with the computer and, implicitly, upon the way of operating with the mental model can be ascribed to anxiety directed specifically toward computers. This latter has strong situational features. Nevertheless, if supported by a background of general anxious behavior, it may constitute a factor rather hard to change.

Individuals showing high anxiety (as a trait) and those who usually hold in or suppress their angry feelings manifest lack of assurance in operating the computer and are afraid they might commit errors. They don't operate with icons.

Persons who express anger toward other people or objects in their environment have an inferiority complex with respect to the computer. They show a prediction for operating with icons, often work with a set of production rules, and do not use programs in their mental representation of the SPSS product.

Users characterized by a bent for dissimulation in the sense of "fake good" prefer to interact with the computer through verbal descriptions, and those with high neuroticism scores reject communication based upon graphic symbols (visual image).

(3) The level of knowledge in the field concerned by the software application influences merely the correctness of the mental representation of the software in point.

(4) Anxiety as a trait influences in a positive sense the completeness of the mental image.

(5) Computer anxiety can be nourished by some cultural or educational clichés. Under such circumstances it can be alleviated by means of appropriate educational measures. We assume that such educational means are apt to change those dimensions of computer anxiety related to:

- the user's belief that he is unable to learn computer work,
- his conviction that not everybody can learn to use a computer, and
- his failing to acknowledge the importance of computers in the work process.

(6) The components of the mental model are not dependent on sex: they are found to the same degree in males and females.

It appears that computer anxiety is a behavior pattern which, if grafted upon an anxious personality structure, lessens substantially the chances of

constructing an efficient mental model of communication with the computer.
When computer anxiety is only a situational peculiarity, multiple possibilities of intervention are available for the designer, so the problem can be overcome in time. Of course, further depth research is indispensable.

Acknowledgements
The work reported in this paper has been supported by grants from the Commission of the European Comunities (Contract No. CIPA3510CT924603). I want to thank in particular to Professor C.R. Cavonius Ph.D. from the Institut für Arbeitsphysiologie an der Universität Dortmund, who encouraged and offer me valuable comments of this work. A special thank to colleagues in the Abteilung Sinnes- und Neurophysiologie of the Institut für Arbeitsphysiologie Dortmund.

REFERENCES

Brazier, F.M.T. (1991): *Design and Evaluation of a User Interface for Information Retrieva*l. Vrije Universiteit, ICG Printing Dordrecht.

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

Eysenck, H.J. & Eysenck, S.B.G. (1975): *Manual for the Eysenck Personality Questionnaire*. Hodder & Stoughton Educational.

Eysenck, S.B.G., Băban, A., Derevenco, P. & Pitariu, H. (1989): A cross-cultural study of personality: Rumanian and English adults. *Revue Roumaine des Sciences Sociales. Serie de Psychology*, 33, 1, 75-80.

Heinessen, R.K., Jr., Glass, C.R. & Knight, L.A. (1987): Assessing computer anxiety: Development and validation of the computer anxiety rating. *Computers in Human Behavior*, 3, 49-59.

Kahana-Moscu, I. & Vrînceanu, M. (1983). Un test de anxietate trades şi validat în limba română. *Revista de psihologie*, 29, 1, 36-48.

Moran, T.P. (1981): The Command Language Grammar: a representation for the user interface of interactive computer systems. *International Journal of Man-Machine Studies*, 15, 3-50.

Norman, D.D. (1986): Cognitive engineering. În D.A. Norman,& S.W. Draper (Eds.), *User Centered Systems Design*. N.J: Lawrence Erlbaum, Hillsdale.

Opperman, R. (1990): Experiences with evaluation methods for human-computer interaction. *Proceedings of the ECCE - Fifth European Conference on Cognitive Ergonomics*.

Pask, G. & Scott, B.C.E. (1972): Learning strategies and individual competence. *International Journal of Man-Machine Studies*, 4, 217-253.

Preece, J., Rogers, Y., Sharp, H., Benyon, D., Holland, S. & Carey, T. (1994). *Human Computer Interaction*. Wokingham: Addison-Wesley Publishing Company.

Shneiderman, B. (1979): Human factors experiments in designing interactive systems. *Computer*, 12, 9-24.

Spielberger, Ch.D. (1988): *State-Trait Anger Expression Inventory*. Psychological Assessment Resources, Inc., Odessa, Florida.

Spielberger, Ch.D., Gorsuch, R.L. & Lushene, R.E. (1970): *STAI.* Consulting Psychologists Press, Inc. Palo Alto, Ca.

Van der Veer, G. C. (1990) *Human Computer Interaction. Learning, Individual Differences and Design Recommendations*. Offsetdrukkerij Haveka B.V., Alblasserdam.

Van der Veer, G.C. (1989). Individual Differences and the User Interface. *Ergonomics*, 32, 11, 1431-1449.

Quality Models for Automated Evaluation of Web Sites Usability and Accessibility

Francisco Montero[1,2], Pascual González[1], María Lozano[1], Jean Vanderdonckt[2]

[1]Grupo de Investigación LoUISE, EPSA – UCLM, 02071 Albacete, Spain
{fmontero, mlozano, pgonzalez}@info-ab.uclm.es

[2]Belgian Lab. of Computer-Human Interaction – UCL, 1348 Louvain-la-Neuve, Belgium
{montero,vanderdonckt}@isys.ucl.ac.be

ABSTRACT
When usability evaluation is performed on web sites, many different evaluation methods can be used that are analytical or empirical, depending if they are conducted with or without end users, on a real web site (part or whole) or on a representation of it. A classification on evaluation methods is given on these parameters so as to assess the relevance and appropriateness of each evaluation method. For this purpose, the quality models for web usability evaluation need to be characterized. When a method is applied (semi-) automatically on a web site, the characterization of these quality models become even more preeminent and crucial to really know the relevance and appropriateness of the results provided by the automated method. Towards this end, different quality models are compared based on guideline review to show their various levels of precision, their advantages and shortcomings

Author Keywords
Accessibility, automated evaluation, criteria, evaluation method, factor, guidelines, metric, quality model, quality of web sites, usability.

ACM Classification Keywords
H.5.2.e Evaluation/methodology and H.5.2.n Style guides.

INTRODUCTION

Despite the abundance of usability and accessibility knowledge, web site quality continues to be a pressing human-computer interaction issue. The majority of web sites have usability and/or accessibility problems, which can result in confusing users, and ultimately, loss of revenue. One of biggest signs that a site has quality problems is that users struggle to find the information they are looking for or they simply cannot manipulate the contents as they wish. Some web pages are so cluttered that users can easily miss the link or the feature that they are looking for. Given that an estimated 90% of web sites provide inadequate usability, a projected growth of 196 million new sites within the next five years, and a severe shortage of user interface professionals to ensure usable sites: tools and methodologies are needed to accelerate and improve the web site design process [3]. Additionally, not every organization can afford to spend millions to hire professionals to design their sites.

To achieve software quality in a system, the software's attributes must be clearly defined. Otherwise, assessment of quality is left to the intuition or the responsibility to the persons who are in charge of the process. In this sense, a quality model must be built and evaluation methods should be used during design and implementation stages based on these quality models. These web quality models can take many different forms depending on the emphasis unconsciously or consciously put on some part:

- Software Engineering pays a lot of attention to the software quality in terms of factors such as correctness, robustness, extendibility, and reusability.
- Software Performance tends to privilege performance factors such as rapidity, compactness, and efficiency.
- Assistive technologies consider that the most important quality factor of a web site resides in its accessibility as any web site should be accessed by the widest audience possible (perhaps with different disabilities or specific needs), in different contexts of use, with various computing platforms.
- Mobile Computing is interested with factors affecting the code and the contents of a web site so as to transcode it to a mobile phone, a Personal Digital Assistant (PDA), a handbag PC, a tablet PC or a laptop.
- The area of Graphic Design includes skills and requirements for ensuring the quality of the visual design of web sites.
- Web Engineering applies systematic, disciplined and quantifiable approaches to development, operation, and maintenance of Web-based applications. Not

only developers are busy with the well-formedness of code with respect to established standards, but also the validity of the code is an integral part of the quality.

- Last, but not least, Human-Computer Interaction has tackled how to assess the usability and the accessibility of a web site according to user-centered methods.

Since there is no universal method for measuring the usability and the accessibility of a web site in all circumstances that guarantees the representativeness of these notions, many attempts exist to approximate usability and accessibility by different models, involving different types of functions. As quality, usability, and accessibility are concepts requiring interpretation, we assign the following goals to this paper:

- We must identify the underlying quality models used for (semi-) automated evaluation of web sites for usability and accessibility. For example, the ISO 9126 [16] standard decomposes quality into six factors (*functionality, reliability, efficiency, portability, usability, maintainability*) that are in turn decomposed into sub-factors, for instance usability contains *learnability*, *operability* and *understandability*. The standard does not prescribe how these sub-factors can be effectively and efficiently measured.
- We need to identify what are the potential shortcomings of usability and accessibility evaluation performed on these models. For example, web designers, developers and evaluators need to understand general characteristics of web quality evaluation tools as they need to know that there is more to ensuring the quality of a website than merely checking it with a tool.
- We need to identify the strengths and weaknesses of tools for automated evaluation. For example, many accessibility tools are based on accessibility guidelines, e.g. WCAG [24] or Section 508 [22], but vary slightly in their interpretation of guidelines. So these results still require human judgment.

To address the three above goals, this paper first surveys some automated or semi-automated tools for usability and accessibility evaluation. This section ends up with a table comparing the salient characteristics of these tools so as to identify their strengths and weaknesses. Then, a framework is introduced to identify and express their underlying quality models that are further analyzed and discussed in the fourth section.

RELATED WORK

Several surveys of evaluation methods for user interfaces exist: Hom [15], Zhang, [25] or Ivory [17], provide a detailed discussion of inspection, inquiry or testing methods and automatic tools. Ivory introduced a taxonomy for classifying evaluation methods [17]. She groups methods along four dimensions: method class, method type, automation type and effort level. Guidelines and automatic testing tools for websites play an important role in determining a set of attributes and measurement methods that are both viable and reliable. However, guidelines and tools need to be validated. A general classification of guidelines that can be used to position a tool for Web evaluation by guideline review has also been introduced [18] in which guidelines are considered during operationalization and evaluation.

Automatic tools for websites analysis can be used to inspect the source code, to inspect the live web pages, to inspect the web server logs of usage of a website, to test the performance of the web server and back ends, to test the positioning of a site on search engines. Automatic tools are able to detect only features related to internal attributes; there's no way for them, in a totally automatic way, to determine external attributes. Automatic evaluation tools are potentially beneficial in the web site design process [11].

Related work in software engineering

Pioneering work on quality factors was performed by McCall [19] and by Boehm [7]. These early models are usually criticized because they lack rationales used to decide which factors should be included in the definition of the quality and which criteria should associated with specific factors. In addition, the metrics used in the bottom layer are not specified, which makes the application of these models difficult in practice. McCall's model includes eleven quality factors to describe three aspects of the product quality: operation, revision, transition. Boehm model uses seven quality factors to evaluate the maintainability and utility.

The ISO 9126-1 [16] model improves the McCall model by defining six factors but does not elaborate on criteria and metrics layers. All these models suppose the quality factors are higher level and not easily directly measured.

Related Work in Web Engineering

Quality of web sites and their associated factors like usability, accessibility or performance are very interesting, and there are many ongoing research and development projects in this scope. WebTango [17], WebQEM [20] or Kwaresmi [5] are only a few relevant examples in evaluating quality on the web. So, before introducing our proposal in this issue, we would like to comment briefly some of these meaningful previous proposals. So, let us focus now on the usability factor of quality. A summary table of these proposals and tools is shown in Table 1.

WebTango of Ivory [17] presents a synthesis of usability and performance evaluation techniques, which together build an empirical foundation for automated interface

evaluation. WebTango is able to capture up to 157 different metrics of each web site, depending on its type. However, only six of them are effectively used to assess the usability of a web site in the underlying model: the actual value of each of this metric is then compared with a reference value of this metric depending on the site type. The reference values have been previously empirically validated by several user testing experiments on different types.

The general approach of WebTango [17] involves:

1. Identifying an exhaustive set of quantitative interface measures, building a quality model. Under nine characteristics, 157 highly-accurate, quantitative page-level and site-level measures are proposed. The measures assess many aspects of Web interfaces (Eq. 1). Each criteria has associated several quantitative metrics, these are related with text, link, and graphic elements or formatting, page formatting or performance and site architecture measures. Ivory proposes a **statistical quality model** (Eq. 1);
2. computing measures for a large sample of rated interfaces;
3. deriving statistical models from the measures and ratings;
4. using the models to predict ratings for new interfaces, and
5. validating model predictions.

$$\mathbf{U}(p) \approx f_{\text{WebTango}}(\textit{content, structure and navigation, visual design, functionality, interactivity, overall experience}) \quad \text{(Eq.1)}$$

Olsina [20] follows common practice in describing software quality in terms of quality characteristics as defined in the ISO/IEC 9126-1 [16] standard. **WebQEM** [20] starts from the ISO model and customizes it at the sub-factor level. WebQEM sees attributes as measurable properties of an entity and propose using a **linear quality model** (Eq. 2) following linear additive and non-linear multimedia scoring criteria to specify them.

$$\mathbf{U}(p) \approx f_{WebQEM}(\text{ISO 9126-1 customizing it at the subcharacteristics level}) = \\ = \sum_{i=1}^{attributes} a_i x_i + \sum_{j}^{attributes} \left(b_j y_j^r \right)^{1/r} \quad \text{(Eq. 2)}$$

where a_i and b_j are weight and x_i and y_j are elemental indicators

Kwaresmi [5] is a framework that defines a systematic and consistent way for structuring guidelines in order to enable their automatic evaluation; a Guideline Definition Language (GDL) able to express guideline information in a sufficiently rich manner to enable an evaluation engine to perform automated evaluation of any GDL-compliant guideline and a tool to support the proposal.

$$\mathbf{U}(p) \approx f_{kwaresmi}(\text{Web\_page}, \text{UES}_{i,j}) = \\ \mathbf{EXEC}\ (\text{EC}_{i,j}\ \{\text{INST\_UES}_{i,j}\}) = \\ \{\text{“}\textit{Respected}\text{”} \mid \text{“}\textit{Violated}\text{”} \mid \text{“}\textit{Partially Respected}\text{”}\} \quad \text{(Eq. 3)}$$

where $\text{UES}_{i,j}$ be the set of evaluation sets associated to the guideline i in the source j and that will be used for the evaluation of the evaluated page. $\text{EC}_{i,j}$ be the set of evaluation conditions associated to $\text{UES}_{i,j}$. $\text{INST\_UES}_{i,j}$ be the set of captured instances of $\text{UES}_{i,j}$ in the evaluated page

In practice, the $f(\text{Web\_page}, \text{UES}_{i,j})$ (Eq. 3) executes each $\text{EC}_{i,j}$ condition, and then it combines the results to have the overall result for the guideline i. We say that a web page satisfies a guideline $G_{i,j}$, if the execution of all $\text{EC}_{i,j}$ on all the $\text{INST\_UES}_{i,j}$ is true. Using the above evaluation parameters allows us to define a kind of quality model to balance the evaluation result. Contrary to the binary model used by most existing evaluation tools, Beirekdar uses a weight concept to express the evaluation result. Kwaresmi uses a **linear quality model**.

In the accessibility field, **Bobby** [6], **Valet** [2] and **EvalIris** [1, 13] are representative examples of accessible evaluation tools. All these tools are based on accessibility guidelines, so for example Bobby helps authors determine if their sites are accessible. It does this through automatic checks as well as manual checks. It also analyzes web pages for compatibility with various browsers (Eq. 4). Accessibility tools use a **binary model** to evaluate the accessibility of web pages (Eq. 4)

$$\textit{Accessibility errors} = \sum_{i=1}^{guidelines} a_i x_i \quad \text{(Eq. 4)}$$

where a_i is 0 when guideline is violated and 1 when guideline is not violated and x_i is a guideline.

Tool	WebQEM	WebTango	Kwaresmi	EvalIris	Valet	Weblint
Author	Olsina [20]	Ivory [17]	Beirekdar [5]	Abascal [1, 13]	WebThing Ltd. [2]	Bowers [8]
Application	Quality	Usability	Accessibility	Accessibility	Accessibility	HTML validator
Model	Linear and non-linear	Statistical	Linear	Binary	Binary	Binary
Scope of approach[3]	Product	Product	Product	Product	Product	Product
Defines factor[4]	Yes	No	No	No	No	No
Defines criteria[4]	Yes	No	No	No	No	No
Defines sub-criteria[4]	Yes	Yes	Yes	No	No	No
Suggests metrics[4]	No	Yes	No	No	No	No
Aspect measured	Metrics	Metrics	Errors	Errors	Errors	Errors
Point of view	Dev.[1] / User	Dev. /User	Dev.	Dev.	Dev.	Dev.
Interpreting data	Quantitative	Qualitative	Qualitative	Quantitative	Quantitative	Quantitative
Evaluation technique	Semi-autom. inspection	Automated inspection	Automatted inspection	Automated inspection	Automated inspection	Automated inspection
Implementation	Hard coded[2]	Hard coded	GDL	XML	Hard coded	Hard coded
Based on	ISO 9126	Webby score	Guidelines	Accessibility guidelines	Accessibility guiielines	HTML specification

Table 1. Some meaningful usability and accessibility evaluation methods and tools

[1] Dev ≈developer's point of view

[2] All *hard coded* means is that it is constant and very hard to change.

[3] Each method or tool can be product or process-oriented. All proposes are product-oriented.

[4] These rows (*defines factor, criteria, sub-criteria and metrics*) refer to the possibility of using these elements as building blocks for quality model construction

A FRAMEWORK FOR QUALITY MODELS IN WEB AUTOMATED EVALUATION

Consider the examples presented in the previous section, we can see that the quality of a web site is a property difficult to define and capture in an operational way. We observe two meaningful things: there is not a universal quality model and there is a little relationship between quality errors and how to prevent them. To improve the understanding of these shortcomings, a reference framework for quality models in web automated evaluation is introduced (Fig. 1) that decomposes the quality into several factors and links them using experience in form of guidelines.

A ***quality model*** specifies which properties are important for ensuring the quality of a web site [9, 10], it is a description of which criteria are important for the analysis, which one is more important than others, and which measurement methods have to be used to assess the criteria. Quality may depend on task-related factors, performance-related factors and development-related factors. A ***factor*** is a statement of a general evaluation dimension which is expressed symptomatically by intrinsic qualities and deficiencies and which could be measured and/or estimated.

For instance, the ISO 9126 [16] definition of quality for software products is *the totality of features and characteristics of a software product that bear on its ability to satisfy stated or implied needs*. In this definition, quality is decomposed into six factors which are each in turn further decomposed into *criteria*. ***Criteria*** are recognized and accepted dimensions that are empirically proved to influence the quality. For instance, usability is decomposed in ISO 9126 into three criteria: *learnability*, *operability*, and *understandability*.

Alternatively in the ISO 9241 standard, usability is decomposed into *effectiveness*, *efficiency*, and *user satisfaction*. We are focus on usability and in [21], usability is decomposed into 8 ergonomic criteria: *compatibility*, *consistency*, *workload*, *adaptability*, *dialog control*, *representativeness*, *guidance*, and *error management*. *Ergonomic criteria* are primarily considered as design criteria because they can serve at design time, but they can also serve as evaluation criteria at evaluation time. For instance, it is interesting to see the impact of ergonomic criteria (e.g., *consistency*) on factors (e.g., *usability*). When appropriate, criteria can be recursively decomposed into a taxonomy of sub-criteria. For instance, consistency can be refined into consistency of location, of presentation formats, of dialogue (Fig. 1).

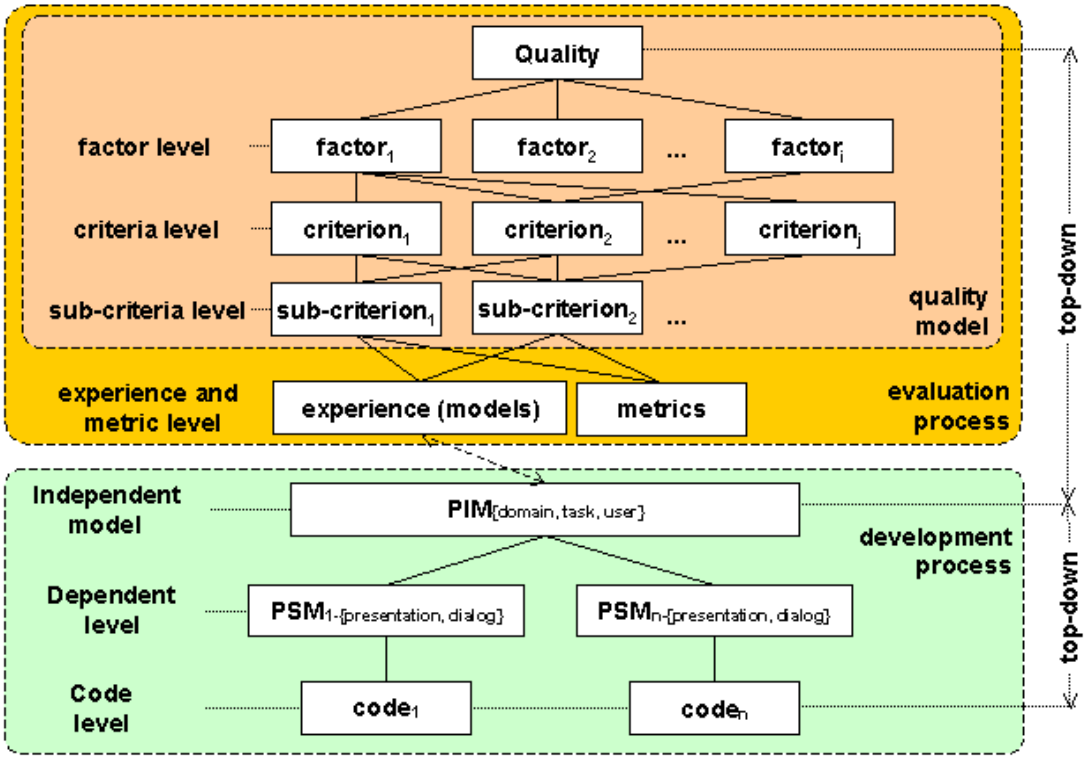


Figure 1. Reference framework for automated evaluation of web sites

Our reference framework is aimed at providing explicit guidance in measuring and applying the criteria. For this purpose, the concept of experience is introduced (Fig. 1) in the definition of our quality model. ***Experiences*** are advices on web design based on good practices which have web design professionals. This definition highlights that following experience is a necessary but sufficient condition to reach the goal. In this framework experience is modeled using different notations and this experience is platform-independent and a same experience can participate in several criteria, thus introducing a multi-criteria approach. For instance, to promote the criteria of consistency, experience in form of models can be manipulated that govern the consistency within a web page or across web pages, for every type of contents such as objects, menus, icons, controls, their locations, their presentation, and their dialogs.

To concretely assess the verification of these models, one or several metrics are required. In our framework, a ***metric*** is a quantifiable dimension that explicitly participates into the verification of a guideline. For instance, to check the guideline "Location of controls should be consistent", the different controls need to be identified and their location, computed. Then, the guideline may be assessed in some automated way. An *object* is any widget composing the user interface of a web page. This includes banners, images, illustrations, videos, controls (e.g., edit boxes, radio buttons, push buttons), panes, etc. Pages when assembled form the web site submitted to evaluation or a presentation of it, typically a significant subset of it.

Consequently, this framework combines two top-down approaches, in one of them quality is progressively refined into models and metrics, and in the other approach those models are used to specify a software product where quality criteria of interest for the evaluation are explicitly considered.

This framework is inspired in different experiments and tendencies. First of all, usability criteria need to be integrated into web site development training. In addition, usability criteria need to be adapted such that designers, who do not have a background in human factors, can apply them. In this sense, there are different experiments [12] where is showed that a student group identified 51% more usability problems in the condition with the usability criteria than in the conditions without guidelines or with the ISO/DIS 9241-10 standard. In these experiments, participants were familiar with ergonomic and usability aspects (they were in a cognitive ergonomics university program), but in these experiments was determined that people, who do not have a background in ergonomics or human factors, can similarly apply usability criteria. So in our framework, in the criteria level ergonomic criteria are considered. These ergonomic criteria are related with quality factors from quality and usability international standards (i.e. ISO 9126, ISO 9241).

Second, in this moment Model-Driven Architecture (MDA) is the tendency in software development. The MDA defines an approach to modeling that separates the specification of system functionality from the specification of its implementation on a specific technology platform. In short it defines guidelines for structuring specifications expressed as ***models***. The MDA promotes an approach where the same model specifying system functionality can be realized on multiple platforms through auxiliary mapping standards, or through point mappings to specific platforms. It also supports the concept of explicitly relating the models of different applications, enabling integration and interoperability and supporting system evolution as platform technologies come and go.

Third, in many occasions quality is additional non-explicit functionality (copy, paste, undo, redo, feedback, error management, help, assistance, information, etc.). This functionality can be modeled and these models can be associated with usability criteria.

With our framework quality model and software development process are linked using models. Those models are platform-independent because users need quality and usability in whatever platform.

So, in this workshop several questions can be introduced, for instance, Can the quality be modeled?, Is the quality platform-independent?. In this moment, quality is adequately considered in the software development process? Is the quality a quantitative or a qualitative feature? etc.

CONCLUSION

Quality web applications need to be usable, functional, reliable, maintainable, scalable and secure. A wide range of evaluation techniques have been proposed and a subset of these techniques is currently in common use. In this sense, we need two ingredients: development process and quality evaluation. There is thus a strong need for Web Engineering. We need to get a better understanding of the development process itself, we need to gain a much clearer understanding of the development process and how it relates to the resultant qualities of web sites. Developing Web based systems is significantly different from traditional software development and poses many additional challenges, some of them were introduced in this paper.

Some techniques and tools pretend to assess the usability or the accessibility of web sites, but they actually do not: they replace usability by metrics that are not necessarily related to usability. Some others replace usability by a mix of usability-oriented metrics and a set of non-usability-oriented metrics. Sometimes, these metrics are not directly related to usability, but they may influence it. For example, the time to download a web page considered per se is a performance factor than a usability factor. But when one knows that usability guidelines recommend that the system response time stays within the limits of 2 sec for a simple

task and 5 sec for a more complex task, then the downloading time becomes interesting. The problem is that when this situation occurs, there is little or no guidance on how to turn the computed metrics into interpretable data.

In this paper we introduced a framework where experience and web site development are linked using qualitative good practices. This experience is modeled and is organized using ergonomic criteria These models can be used in software analysis and design stages. Using this experience novice programmers can find usability problems and they can do changes in the specification of their software products to achieve quality improvements in theirs products.

ACKNOWLEDGMENTS

This work was supported by the Spanish CICYT projects TIN2004-08000-C03-01 and PBC-03-003. We gratefully acknowledge the support of the SIMILAR network of excellence (www.similar.cc) and the COST European Action n°294 (MAUSE).

REFERENCES

1. Abascal J., Arrue M., Fajardo I., Garay N., Tomás J.: Use of Guidelines to Automatically Verify Web Accessibility. Universal Access in the Information Society (2004)
2. Valet. http://valet.webthing.com/access/url.html
3. Barry, C., Lang, M.: A survey of multimedia and web development techniques and methodology usage. IEEE 8, 2 (2001)
4. Basili, V., Caldiera, G., Dieter H.: The Goal Question Metric Approach. In: Caldiera, G., Rombach, D.H. (eds.): Encyclopedia of Software Engineering. John Wiley: New York, (1994)
5. Beirekdar, A., Vanderdonckt, J., Noirhomme-Fraiture, M.: A Framework and a Language for Usability Automatic Evaluation of Web Sites by Static Analysis of HTML Source Code. In: Kolski, Ch. Vanderdonckt, J. (eds.): Proc. of 4th Int. Conf. on Computer-Aided Design of User Interfaces CADUI (2002)
6. Bobby. http://webxact.watchfire.com/
7. Boehm, B.W., Brown, J.R., Kaspar, H., Lipow, M., Macleod, G.J., Merritt, M.J.: Characteristics of Software Quality. North-Holland: Amsterdam (1978)
8. Bowers, N. Weblint: Quality Assurance for the World-Wide Web. Journal reference: *Computer Networks and ISDN Systems*, Volume 28, issues 7–11, p. 1283. (1996)
9. Brajnik, G.: Towards Valid Quality Models for Websites. In: Proc. of 7th Conf. on Human Factors and the Web HFWeb'01 (2001)
10. Brajnik, G.: Quality Models based on Automatic Webtesting. In: Proc. of the CHI'2002 Workshop "Automatically Evaluating Usability of Web Sites" (2002)
11. Brajnik, G.: Comparing Accessibility Evaluation Tools: Results from a Case Study. In: Ardissono, L., Goy, A. (eds.), Proc. of Symposium on Human-Computer Interaction HCITALY'2003 (2003)
12. Chevalier, A., Ivory, M. Can novice designers apply usability criteria and recommendations to make web sites easier to use ? , in Proc. of Int. Conf. on Human-Computer Interaction HCI International. (2003)
13. EvalIris. http:// www.sc.ehu.es/acwbbpke/evaliris.html
14. Farenc, Ch., Palanque, Ph., Vanderdonckt, J., User Interface Evaluation: is it Ever Usable?, in Proc. of 6th Int. Conf. on Human-Computer Interaction HCI International. (1995).
15. Hom, J.: The Usability Methods Toolbox. http://jthom.best.vwh.net/usability/ usable.htm. (1996)
16. ISO/IEC 9126-1. Software Engineering–Product Quality–Part 1: Quality Model. International Organization for Standardization: Geneva (2001)
17. Ivory, M.: Automated Web Site Evaluation. Researchers' and practitioners' perspectives. Human-Computer Interaction Series, Vol. 4, Kluwer Academic Pub.: Dordrecht (2003)
18. Mariage, C., Vanderdonckt, J., Pribeanu, C.: State of the Art of Web Usability Guidelines. In: Proctor, R.W., Vu, K.-Ph.L. (eds.): The Handbook of Human Factors in Web Design. (2004)
19. McCall, J.A., Richards, P.G., Walters, G.F. Factors in Software Quality, Vols. I, II, and III. NTIS. (1977)
20. Olsina, L., Rossi, G.. Measuring Web Application Quality with WebQEM. IEEE Multimedia (2002)
21. Scapin, D.L., Leulier, C., Vanderdonckt, J., Mariage, C., Bastien, Ch., Farenc, Ch., Palanque, Ph., Bastide, R.: A Framework for Organizing Web Usability Guidelines. In Kortum, Ph., Kudzinger, E. (eds.): Proc. of Conf. on Human Factors and the Web HFWeb'2000 (2000).
22. United States Rehabilitation Act Section 508 standard. http://www.section508.gov
23. Vanderdonckt, J., Beirekdar, A., Noirhomme-Fraiture, M., Automated. Evaluation of Web Usability and Accessibility by Guideline Review, Proc. of 4th Int. Conf. on Web Engineering ICWE'04. (2004)
24. World Wide Web Consortium. Evaluating Web Sites for Accessibility. http://www.w3c.com
25. Zhang, Z., Basili, V., and Shneiderman, B.: An empirical study of perspective-based usability inspection. Human Factors and Ergonomics Society Annual Meeting, Chicago. (1998)

Quality Models Involved in Automated Evaluation of Web Usability and Accessibility Guidelines

Abdo Beirekdar[1], Jean Vanderdonckt[2], Monique Noirhomme[1]

[1]Facultés Universitaires Notre-Dame de la Paix
rue Grandgagnage, 21
B-5000 Namur, Belgium
{abe, mno}@info.fundp.ac.be

[2]Université catholique de Louvain
Place des Doyens, 1
B-1348 Louvain-la-Neuve
vanderdonckt@isys.ucl.ac.be

ABSTRACT

Based on the concepts of evaluation sets and conditions, a technique and software are presented that automatically evaluate Web pages by static analysis of their HTML code against usability guidelines. It relies on separating guidelines evaluation logic from the evaluation tool. Due to this separation, the whole evaluation process can be divided into three steps: guideline structuring (specification of evaluation logic), page parsing, and evaluation. A Guideline Definition Language (GDL) is used to specify the formal structure of the guidelines. A tool supports the simultaneous evaluation of multiple guidelines selected on demand from different sources. It also optimizes evaluation by automatically identifying common sub-structures among structured guidelines. Evaluators with different usability practices can express alternative evaluation strategies supported by the tool flexibility. Quality models involved in such an automated evaluation are defined and exemplified.

ACM Classification Keywords

D.2.2 **[Software Engineering]:** Design Tools and Techniques – *user interfaces*. H.1.2 **[Models and Principles**]: User/Machine Systems. H.5.2 **[Information Interfaces and Presentation]:** User Interfaces –*Prototyping, Graphical User Interfaces (GUI).*

General terms: Design, Languages, Human Factors.

Author Keywords

Accessibility guidelines, automated evaluation, quality models, static analysis, usability guidelines.

1. INTRODUCTION

Usability and Accessibility (U&A) are today recognized as critical factors of success for web sites. A wide range of U&A evaluation techniques has been proposed and many of them are currently in use [9,12]. They range from informal usability testing to formal usability tests conducted by usability specialists in usability labs or with real users.

Automation of these techniques became much desired [3,4, 11,15] because they required usability specialists to conduct them or to analyze evaluation results, which is very resource consuming especially for large, continuously growing web sites. In addition, there is a lack of U&A experts due to an increased demand. A possible solution consists of capturing the knowledge and experience of these experts and expressing it in form of recommendations or guidelines to be reviewed and applied by designers and developers. Some studies show that applying guidelines by designers is subject to interpretation, basically because of the inappropriate structuring or formulation [17,19].

For this reason and others, automation has been predominately used to objectively check guideline conformance or review [11]. Many automatic evaluation tools were developed to assist evaluators with guidelines review by automatically detecting and reporting ergonomic violation and making suggestions for repairing them. Some tools are integrated with popular web design tools and methods [18], like Web content Accessibility Guidelines and Section508 guidelines in Macromedia Dreamweaver.

In this paper, we present our approach to overcome this shortcoming. It enables evaluators or human factors experts to express the ergonomic body of knowledge provided by guidelines in terms of constraints of HTML elements (i.e., tags, attributes). Once coded, this knowledge can be evaluated dynamically at evaluation-time by configuring the guidelines expressions in an optimized way depending on the guidelines to be evaluated and the elements contained in the page. This process consequently considers guidelines relevant to the targeted evaluation context, and factors out sub-structures that are common across these guidelines, even if they come from different sets of guidelines.

This paper is structured as follows: section 2 positions our approach in the field of automated web evaluation. Section 3 presents fundamental concepts of our evaluation approach. Section 4 details steps of the evaluation process to be exemplified in Section 5. Section 6 underlines the possibilities of evaluation optimization. Section 7 introduces the implementation of the tool support the approach. Section 8 defines the quality models used in this automated process, before concluding with some advantages.

2. RELATED WORK

Many evaluation tools were developed to provide automation of all or a part of the evaluation process. Brajnik

[3] classifies them according to:

- *Location*: on-line (web-based) vs. off-line.
- *Type of service*: analysers (they only find and highlight defaults; sometimes ranking the list of defaults according to their severity) vs. analysis and repair tools (they assist the developer also in fixing the defaults).
- *Scope*: i.e. the set of attributes that are considered during the automatic analysis. A classification based on scope includes:
  - HTML validators and cleaners assist developers in removing non standard usage of the language).
  - HTML/graphic optimisers improve downloading and rendering performance by re-coding certain parts of HTML or graphic documents.
  - Link checkers probe all the links leaving a page to determine if their target exists.
  - Usability tools detects and sometimes help to fix usability faults and defects

According to this classification, an evaluation tool adopting our approach would be an off-line and/or on-line usability tool. Right now; the following usability tools have been developed:

- A-Prompt, from University of Toronto [2]; off-line, with ranking; does also repair.
- Bobby from Watchfire [4]; web-based and off-line, with ranking.
- ERGOVAL [5,6], from University of Toulouse I, is an off-line analyser for GUIs.
- GUITester2 [15] is an off-line analyser focusing on measuring the consistency of GUIs, like Sherlock [14].
- LIFT from UsableNet.com [18]; web-based and off-line, with ranking and repair.
- Sherlock [7] is off-line analyser that offers some flexibility in incorporating external assessment procedures from DLL.
- WebCriteria [20] is web-based; comparative evaluation of a website with respect to a benchmark derived from similar well-established web sites.

A common shortcoming of the above tools is that the evaluation logic is hard coded in the evaluation engine, which makes them very inflexible for any modification of the evaluation logic or any introduction of new guidelines. In addition, many of them do not offer much possibilities of controlling the evaluation process like choosing which guideline to evaluate, the level of evaluation at evaluation time. For instance, Bobby [4] only provides the choice of the guidelines set to evaluate: W3C or Section508. In another base of U&A guidelines should be used instead, it is not possible to rely on Bobby. If another interpretation of the guidelines supported by Bobby is needed (this sometimes occurs in testing organisations), it is not possible to modify their code to reflect this change.

3. FUNDAMENTAL CONCEPTS

Figure 1 depicts a global view of the fundamental concepts of our approach and their interactions. These concepts form the blocks of a formal Guideline Definition Language (GDL) that we defined to enable a systematic, automated evaluation-oriented and well defined structuring of guidelines. To facilitate their understanding, we apply them on the guideline: "Use a limited number of font types".

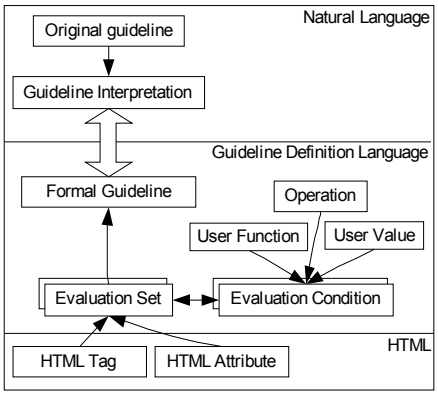


Figure 1. Fundamental concepts.

3.1 Original Guideline

This is the guideline as it is expressed in its source:

```
<Original_GDL id="O1" name="Limited Fonts" source="test"
text="Use a limited number of font types"/>
```

3.2 Interpreted Guideline

As guidelines are generally expressed at a high level of abstraction, the original guideline expressed in natural language should be transformed into a concrete form that can be understood by designers and manipulated by the system. This re-expression is called an *interpretation* of the guideline. In general, interpretation is used to limit the focus of the original guideline to some extent that can be considered satisfactory in the targeted evaluation context. Of course, even with interpretation, evaluation of some guidelines cannot be totally automated [4]. For this reason, every interpretation is assigned to a factor indicating the level of abstraction reflected. In our example, the guideline needs to be interpreted because "limited number" is very abstract. One possible interpretation is:

```
<Inter_GDL id="I1" O_GDL="O1" context="all" S_level="High"
 text="Use less than four fonts"/>
```

3.3 Evaluation set

After interpreting a guideline, we identify HTML elements (tags and attributes) that can be used to evaluate it. These elements are then grouped into evaluation sets. An *evaluation set* usually groups elements that enable the evaluation of a whole or a part of the guideline. Determining the content of a set depends also on the way of specifying associated evaluation condition. Each evaluation set is assigned to a priority attribute to indicate its importance level. We defined three levels that correspond to W3C levels (A, AA and AAA). To evaluate our guideline, HTML provides the attribute FACE of the tag FONT to determine the font family, so, we will need one evaluation set:

```
<Set id="S1" name="" priority="AAA">
```

```
    <Element id="E1" tag="FONT" attribute="Face" scope="ANY"/>
</Evaluation_Set>
```

According to this specification, the parser will capture instances of E1 anywhere in the parsed page. Notice that there is nothing to link S1 to our interpretation. This is because the same evaluation set could be used to structure one or more guidelines. Linking is specified in the formal guideline.

3.4 Evaluation condition

Evaluation conditions correspond to the evaluation logic that must be applied on evaluation sets to evaluate the guideline. Conditions are specified using evaluation sets' components, logical operations (AND, OR, NOT) and some predefined operations (SET_HAS_ELEMENT, EXISTS, NBR_INSTANCES). "User function" and "User Value" concepts shown in Figure 1 enable the evaluator to define new functions and complex values (like a list of numbers) to be used in evaluation conditions. A global condition is formed of smaller conditions called *basic conditions*. These basic conditions provide a mechanism for identifying potential common parts among evaluation conditions. Basic conditions are assigned a priority indicator to avoid ambiguity and to facilitate the execution of conditions. A condition that can be associated to the evaluation set of our example could be the following: NBR_INSTANCES(S1)<=4. Its Guideline Definition Language (GDL) form is:

```
<Condition set="S1">
  <Basic_Cond id="BC1" Operation="NBR_INSTANCES"
        return="number" priority="1">
    <Arg type="Set" value="S1"/>
  </Basic_Cond>
  <Basic_Cond id="" Operation="<" priority="2">
    <Arg type="Basic_Cond" value="BC1"/>
    <Arg type="number"  value="4"/>
  </Basic_Cond>
</Condition>
```

Basic conditions are executed according to their priority. So, we start by finding the number of set instances, then we test if it is smaller than 4. In the detailed example, that kind of specification of evaluation conditions is subject to a complexity due to the desire to provide well structured specification that can easily underline common parts (to foster optimisation) among evaluation conditions.

3.5 Formal Guideline

Formal guideline regroups evaluation sets to be used during evaluation and specifies the order of their evaluation. The default formal guideline associated with a guideline has all sets, and their order of evaluation is their priority level. The formal guideline of our example is:

```
<Formal_GDL id="F1" associated_gdl="I1">
  <Content sets="S1" evaluation_order="default"/>
</Formal_GDL>
```

4. EVALUATION PROCESS

Figure 2 shows the steps of an evaluation process based on our approach. These steps are totally independent, which gives many optimization possibilities at each of them. Of course, there is an order for accomplishing them, and a step can be accomplished at any time as soon as its input resources become available.

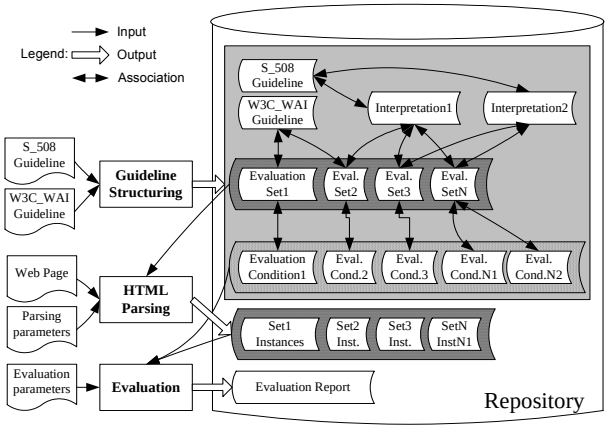


Figure 2. Evaluation process.

4.1 Guideline Structuring

The first step in our process consists of structuring of guidelines in terms of evaluation sets and associated evaluation conditions. Obviously, structuring is highly influenced by the understanding of the original guideline semantics. In addition, it requires good HTML knowledge to identify tags and attributes that can be used to form evaluation sets.

4.2 HTML Parsing

Our evaluation approach is currently limited to pure HTML pages (no CSS). The parsing process is done by a single forward scan of the evaluated page. During this scan, the parser captures the instances of the evaluation sets specified in the formal guideline. We can specify many parsing parameters to control this process: which guidelines to evaluate, number of desired instances (<=N, all), keep or not incomplete set instances, etc.

4.3 Evaluation

After parsing the web page, we can apply the evaluation conditions that we defined during guidelines structuring. Every condition is applied on the captured instances of its corresponding set to determine its respect or violation. By this way, a detailed evaluation report can be generated on respected/violated sets, number of detected instances, and percent of respect/violation.

5. A RUNNING EXEMPLE

Here we apply our approach on two reasonably complex guidelines. **GDL1**: Select colours that will make your page easy to read by people with colour blindness [17]. **GDL2**: Web pages shall be designed so that all information conveyed with colour is also available without colour, for example from context or mark up (Section508).

5.1 Guideline Structuring

Guideline 1 cannot be automated per se in a straightforward manner as there is no calculable way to assess to what extend a page is easy to read or not, depending on the users. However, if we refer to the research conducted

by Murch [17], an interpretation **Inter_GDL1** of this guideline can be produced: "The combination between background color and foreground color should belong to the best color combinations or should not belong to the worst color combinations proposed by Murch". This interpretation will not cover all colors because Murch dealt with basic color only, but we will use it for simplification. **GDL2** needs to be interpreted. As the guideline suggests, information conveyed by color can be conveyed using markup. We will consider the following markup tags: Bold <b>, Italic <i>, text size <Font.size> and text font <Font.face>. Thus, the interpretation of **GDL2** would be **Inter_GDL2**: "Web pages shall be designed so that all information conveyed with color is also available using any combination of the above markup elements". This means that, in our evaluation context, **Inter_GDL2** is considered violated even if colored information was conveyed using other means than the above markup tags. For **Inter_GDL1**, we have the following sets:

- **S1**: control of text color over the whole page. S1={Body.bgcolor^{ANY}, Body.text^{ANY}}.

```
<SET id="S1" name="Global colour control" priority="AAA">
  <Element id="E1" tag="Body" Attribute="text" scope="ANY"/>
  <Element id="E2" tag="Body" Attribute="bgcolor" scope="ANY"/>
</SET>
```

- **S2**: control of colour by Body and Font. S2={Body.bgcolor^{ANY}, Font.color$^{Body.bgcolor}$}.
- **S3**: control of colour by Font and Table. S3={Table.bgcolor^{ANY}, Font.color$^{Table.bgcolor}$}.
- **S4**: control of colour by Font and TH. S4={TH.bgcolor^{ANY}, Font.color$^{TH.bgcolor}$}.
- **S5**: control of colour by Font and TR. S5={TR.bgcolor^{ANY}, Font.color$^{TR.bgcolor}$}.
- **S6**: control of colour by Font and TD. S6={TD.bgcolor^{ANY}, Font.color$^{TD.bgcolor}$}.
- **S7**: control of color by Body and TH. S7={TH.bgcolor$^{Body.text}$, Body.text^{ANY}}.
- **S8**: control of color by Body and TR. S8={TR.bgcolor$^{Body.text}$, Body.text^{ANY}}.
- **S9**: control of color by Body and TD. S9={TD.bgcolor$^{Body.text}$, Body.text^{ANY}}.
- **S10**: control of color by Body and Table. S10={Table.bgcolor$^{Body.text}$, Body.text^{ANY}}.
- **S11**: control of links color. S11={Body.bgcolor^{ANY}, Body.link^{ANY}}.
- **S12**: control of active links color. S12={Body.bgcolor^{ANY}, Body.alink^{ANY}}.
- **S13**: control of visited links color. S13={Body.bgcolor^{ANY}, Body.vlink^{ANY}}.

According to our experience with HTML 4.0, these sets cover all the possibilities provided by HTML to manipulate text color, so, we consider that the evaluation of **Inter_ GDL1** (thus **GDL1**) can be totally automated. The evaluation conditions associated with the above sets are very similar. Next we give the detailed GDL specification for one of them (S1) that corresponds to the next pseudo specification:

```
(Body.text IN ListOfGoodColors(Body.bgcolor)) OR
(Body.text NOT IN ListOfBadColors(Body.bgcolor))
```

ListOfGoodColors and ListOfBadColors are two lists of predefined values (colors). As mentioned earlier in this section, we will use Murch color combinations [17]. We specify basic colors as user values:

```
<User_V id="Black"   type="Hexa" val="#000000"/>
<User_V id="White"   type="Hexa" val="#ffffff"/>
<User_V id="Red"     type="Hexa" val="#ff0000"/>
<User_V id="Geen"    type="Hexa" val="#00ff00"/>
<User_V id="Blue"    type="Hexa" val="#0000ff"/>
<User_V id="Cyan"    type="Hexa" val="#00ff"/>
<User_V id="Magenta" type="Hexa" val="#ff00ff"/>
<User_V id="Yellow"  type="Hexa" val="#ffff00"/>
```

Then, we specify lists of values corresponding to good and bad foreground colors for a given background colour:

```
<User_V id="GoodFgBlackBg" type="List" val="White Yellow"/>
<User_V id="BadFgBlackBg" type="List" val="Blue Red Magenta"/>
<User_V id="GoodFgWhiteBg" type="List" val="Blue BlackRed"/>
<User_V id="BadFgWhiteBg" type="List" val="Yellow Cyan"/>
<User_V id="GoodFgRedBg" type="List" val="White Yellow"/>
<User_V id="BadFgRedBg" type="List" val="Magenta Blue"/>
<User_V id="GoodFgGreenBg" type="List" val="Black Blue Red"/>
<User_V id="BadFgGreenBg" type="List" val="Magenta Cyan"/>
<User_V id="GoodFgBlueBg" type="List" val="Green Red"/>
<User_V id="BadFgBlueBg" type="List" val="Red Magenta"/>
<User_V id="GoodFgCyanBg" type="List" val="White Yellow"/>
<User_V id="BadFgCyanBg" type="List" val="Yellow White"/>
<User_V id="GoodFgMagentaBg" type="List" val="White Black"/>
<User_V id="BadFgMagentaBg" type="List" val="Green Red"/>
<User_V id="GoodFgYellowBg" type="List" val="Black Blue"/>
<User_V id="BadFgYellowBg" type="List" val="White Cyan"/>
```

Then we specify the global condition corresponding to the given pseudo specification. One will easily notice that the specification of conditions is relatively long, but this is primarily intended to be processed by a computer agent instead of being read by a human. The specification reflects the execution sequence reproduced in Fig. 3. The execution order is determined by priority level assigned to every basic condition. Cond1:

```
<Condition set="S1">
  <Basic_Cond id="BC1" operation="=" return="Bool" priority="1">
     <Arg type="SetE" value="E1">
     <Arg type="value" value="Black">
  </Basic_Cond>
  <Basic_Cond id="BC2" operation="IN" return="Bool" priority="1">
     <Arg type="SetE" value="E2">
     <Arg type="value" value="GoodFgBlackBg">
  </Basic_Cond>
  <Basic_Cond id="BC3" operation="NOT IN" return="Bool"
           priority="1">
     <Arg type="SetE" value="E2">
     <Arg type="value" value="BadFgBlackBg">
  </Basic_Cond>
  <Basic_Cond id="BC4" operation="OR" return="Bool" priority="2">
     <Arg type="Basic_Cond" value="BC2">
     <Arg type="Basic_Cond" value="BC3">
  </Basic_Cond>
  <Basic_Cond id="" operation="AND" return="Bool" priority="3">
     <Arg type="Basic_Cond" value="BC1">
     <Arg type="Basic_Cond" value="BC4">
  </Basic_Cond></Condition>
```

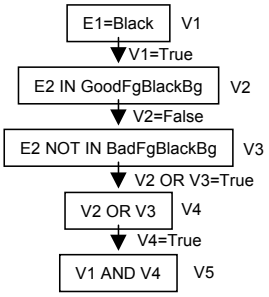


Figure 3. Evaluation sequence.

The above condition is for black background only. We will need to specify conditions for the remaining seven basic colors. We will not do it here because they are almost identical to this condition. For **Inter_GDL2**, we have the following evaluation sets:

– **S1A**: conveying using bold tag.

```
<SET id="S1A" name="bold conveyance" Priority="AAA">
  <Element id="E1" tag="Body" Attribute="bgcolor" scope="ANY"/>
  <Element id="E2" tag="Font" Attribute="color" scope="E1"/>
  <Element id="E3" tag="b" Attribute="" scope="E2"/>
</SET>
```

– **S1B**: conveying using bold tag.

```
<SET id="S1B" name=" bold conveyance" Priority="AAA">
  <Element id="E1" tag="Body" Attribute="bgcolor" scope="ANY"/>
  <Element id="E2" tag="b" Attribute="" scope="E1"/>
  <Element id="E3" tag="Font" Attribute="color" scope="E2"/>
</SET>
```

– **S2A**: conveying using italic tag.

```
<SET id="S2A" name="italic conveyance" Priority="AAA">
  <Element id="E1" tag="Body" Attribute="bgcolor" scope="ANY"/>
  <Element id="E2" tag="Font" Attribute="color" scope="E1"/>
  <Element id="E3" tag="i" Attribute="" scope="E2"/>
</SET>
```

– **S2B**: conveying using italic tag.

```
<SET id="S2B" name=" italic conveyance" Priority="AAA">
  <Element id="E1" tag="Body" Attribute="bgcolor" scope="ANY"/>
  <Element id="E2" tag="i" Attribute="" scope="E1"/>
  <Element id="E3" tag="Font" Attribute="color" scope="E2"/>
</SET>
```

– **S3**: conveying using font face.

```
<SET id="S3" name=" font face conveyance" Priority="AAA">
  <Element id="E1" tag="Body" Attribute="bgcolor" scope="ANY"/>
  <Element id="E2" tag="Font" Attribute="color" scope="E1"/>
  <Element id="E3" tag="Font" Attribute="face" scope="E2"/>
</SET>
```

– **S4**: conveying using font size.

```
<SET id="S4" name=" font size conveyance" Priority="AAA">
  <Element id="E1" tag="Body" Attribute="bgcolor" scope="ANY"/>
  <Element id="E2" tag="Font" Attribute="color" scope="E1"/>
  <Element id="E3" tag="Font" Attribute="size" scope="E2"/>
</SET>
```

Notice that we defined S1A and S1B to cover the case of bold tag. We need to do so because the two expressions <Font color="...">Colored bold text</Font> and <Font color="..."> Colored bold text</Font> give the same visual result. We need to do the thing for italic tag. Following is the evaluation condition for S1A:

```
Exist(Body.text, Font.color, b) AND (Body.text<>Font.color)
```

Exist is a predefined function on set elements. Notice that this condition would not be valid if we did not specify in S3 that we want to capture all instances including incomplete ones. The XML form of this condition would be:

```
<Condition set="S1A">
  <Basic_Cond id="BC1" symbol="Exist" return="Bool" priority="1">
    <Arg type="SetE"          value="E1">
    <Arg type="SetE"          value="E2">
    <Arg type="SetE"          value="b">
  </Basic_Cond>
  <Basic_Cond id="BC2" symbol="<>" return="Bool">
    <Arg type="SetE"          value="E1">
    <Arg type="SetE"          value="E2">
  </Basic_Cond>
  <Basic_Cond id="BC3" symbol="AND" return="Bool" priority="2">
    <Arg type="Basic_Cond"  value="BC1"/>
    <Arg type="Basic_Cond"  value="BC2"/>
  </Basic_Cond>
</Condition>
```

6. GUIDELINE OPTIMIZATION

Decomposing the evaluation process into three steps as described above offers the possibility for optimising evaluation at each of these steps.

6.1 Structuring Step

As parsing Web pages is based on evaluation sets and evaluation conditions defined in this step, we can optimize the evaluation at two levels: for a single guideline, there are two ways: identifying the minimum ensemble of sets needed to evaluate the targeted guideline, and expressing conditions in the most forward way to minimize the number of operations that evaluation engine would need to execute them. At the level of many guidelines, we can optimize evaluation by identifying common structures or sub-structures. This optimization cannot be neglected since guidelines are expressed at a high abstraction level, and as they come from different sources, it is very possible to have guidelines that are totally or partially semantically identical.

6.2 Parsing Step

The first significant optimization at this step is the use of the concept of *exclusion* among evaluation sets. By definition, one evaluation set *Excludes* one (many) other evaluation set(s) if its presence excludes its evaluation. This concept is based on the *Scope* concept related to HTML elements (tags and attributes). Generally, the excluding set has an element whose scope is within the scope of an element of the excluded set. Of course, these two elements must have the same rendering effect.

For example (Fig. 4), in the context of text color evaluation, a set containing the attribute Table.bgcolor (like S1={ Body.text, Table.bgcolor}) excludes a set containing the attribute Body.bgcolor (like S2={Body.bgcolor, Body.text}), because the scope of Table.bgcolor is within the scope of Body.bgcolor. The second optimization is to combine parsing and evaluation in one step. This means that an evaluation condition is triggered as soon as an instance of the associated evaluation set is completely detected in the evaluated Web page. This combination would be optional because, in some situations like the need for a detailed evaluation report, it is desired to capture all instances of evaluation sets (even incomplete or negative ones).

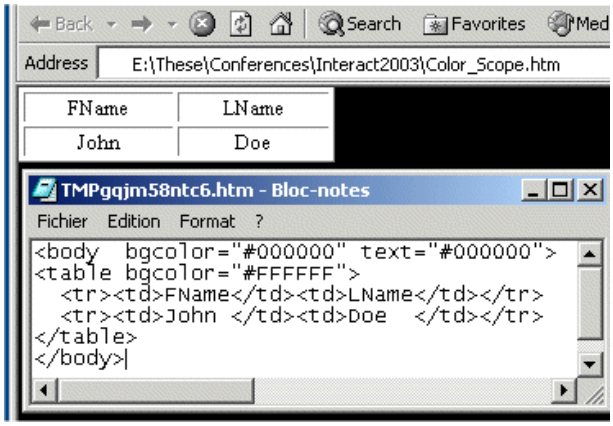


Figure 4: Scope of Table.bgcolor is within the scope of Body.bgcolor

6.3 Evaluation Step

The optimization that can be done at this step relies mainly on optimizing the execution of evaluation conditions. We introduced the concept of *basic condition* to identify similar or identical parts of evaluation conditions. For example, in **Inter_GDL 1** of our example, the evaluation condition for S1 (Cond1) and S2 (Cond2) will be very similar. (same Body.bgcolor). As Cond1 will be executed before Cond2, the results of executing Cond1 can be kept if another instance of Cond2 is met with Font.color having the same value of Body.text. In this way, any evaluation condition is executed once and reused whenever needed so as to factor our.

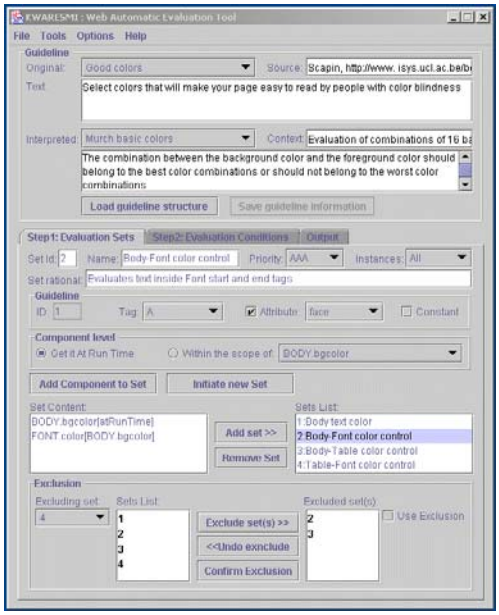


Figure 5: Structuring Step in our automatic evaluation tool.

7. IMPLEMENTATION OF THE EVALUATION TOOL

The specific tool supporting the approach described in this paper is being implemented in Borland Jbuilder V7.0 connected to ODBC data bases. This tool is composed of three modules corresponding to the steps shown in Fig. 2. The structuring module, as already implemented (Fig. 5), was used to structure and store some guidelines in the tool repository. We also developed a simple version of the parsing module. It was tested to evaluate the structured guidelines on web pages of varying complexity. Since each guideline is expressed in GDL, which is a XML-compliant language, it is also possible to display them according to an appropriate style sheet (Fig. 6), thus providing designers with a natural statement of each guideline. First, the natural language expression is given, then the various conditions involved in evaluating the guideline via the evaluation sets.

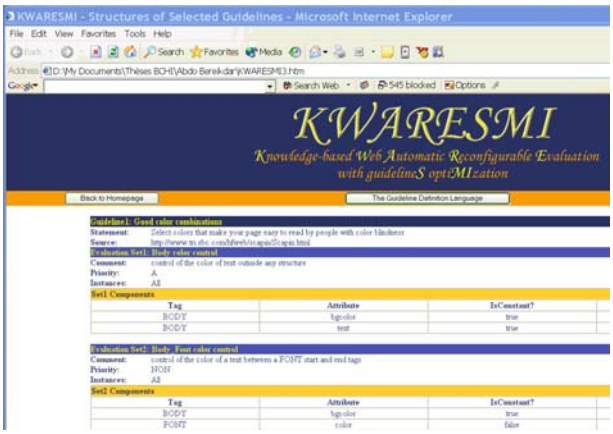


Figure 6. Visualization of a guideline.

8. QUALITY MODELS USED IN AUTOMATED EVALUATION

Using the above evaluation strategy allows us to define a kind of quality model to balance the evaluation result. Contrarily to the binary model used by most existing evaluation tools (a guideline is violated? Yes or No), we can use a weight concept to express the evaluation result. This weight can be predefined as default values associated with the evaluation parameters. And for more flexibility and customization ability, the evaluator should be able to change these values before an evaluation session.

The simplest quality model is the following:

- All ergonomic criteria and page objects have the same weight. Thus, usability or accessibility or other criteria are equally important.
- Priority (1) evaluation sets have a weight of 0.7.
- Priority (2) evaluation sets have a weight of 0.2.
- Priority (3) evaluation sets have a weight of 0.1.

So, let us define the EVAL function in the case of **guideline review**:

Let $nSET_{i,\,j,\,1|2|3}$ be the number of evaluation sets of priority 1|2|3 that must be evaluated.

These sets are the result of considering all the evaluation parameters during the phase of sets identification.

Let $nKO\_SET_{i,\ j,\ 1|2|3}$ be the number of evaluation sets of priority 1|2|3 for which one or more instances were captured in the evaluated page and gave a negative result (violation).

Let $nOK\_SET_{i,\ j,\ 1|2|3}$ be the number of evaluation sets of priority 1|2|3 for which one or more instances were captured in the evaluated page and gave a positive result (respect).

$$nEVAL\_SET_{i,\ j,\ 1|2|3} = nKO\_SET_{i,\ j,\ 1|2|3} + nOK\_SET_{i,\ j,\ 1|2|3}$$

The positive evaluation result will be:

$$Rp = \frac{(nOK\_SETi,\ j,1\ X\ 0.7) + (nOK\_SETi,\ j,2\ X\ 0.3) + (nOK\_SETi,\ j,3\ X\ 0.1)}{(nSETi,\ j,1\ X\ 0.7) + (nSETi,\ j,2\ X\ 0.3) + (nSETi,\ j,3\ X\ 0.1)}$$

The negative evaluation result will be:

$$Rn = \frac{(nKO\_SETi,\ j,1\ X\ 0.7) + (nKO\_SETi,\ j,2\ X\ 0.3) + (nKO\_SETi,\ j,3\ X\ 0.1)}{(nSETi,\ j,1\ X\ 0.7) + (nSETi,\ j,2\ X\ 0.3) + (nSETi,\ j,3\ X\ 0.1)}$$

Such formula allows the simple classification of the evaluated page according to a classification scale. For example, $Rp<0.5$ means bad page, $0.5<=Rp<.75$ means good page, and $Rp>=0.75$ means very good page. Such classification could be enough to publish a list of top N usable sites, or for a webmaster who wants to have a rapid estimation of the quality of designed pages before deciding to go into details about usability problems.

Having all the needed information, the next section will discusses the limits of automation of an evaluation based on the proposed methodology.

9. COST-BENEFITS ANALYSIS

In this section we will present a brief analysis related to the costs and benefits of using the proposed GDL.

9.1 Usability of the GDL

We can consider two criteria related to the GDL usability: human readability and machine "processability".

GDL structures are readable as any XML documents, but are generally long and hard to read by non GDL experts. In addition, we do not pay big attention to this criterion in the context of our work because we think that a GDL editor is needed in any attempt of exploiting our methodology. One purpose of this editor is to facilitate structures readability by presenting them in a convenient manner: graph, tree, etc. and by providing some visualization facilities like zooming, custom hiding/showing of details, etc.

The more important criterion is the ability to process GDL structures, because the interpreter needs to read the structures, extract all needed information and parse Web pages accordingly. This is totally feasible because structures are XML-compliant.

9.2 Users

We can classify potential users of the GDL into the following categories (based on the corresponding tasks):

Guideline Interpreter

The first thing to do is to study the original guideline in order to evaluate the possibility to re-express it more concretely (partially or totally). As we remain at natural language informal level, this task is usually accomplished by HF expert who will give the new expression or leave the guideline as it is if he judges that it is already concrete enough or, on the contrary, it can not re-expressed because it deals with abstract aspects. We are not speaking here about the quality and the reliability of the guidelines because we consider that the HF expert is responsible for this verification. In addition, we do not exclude the possibility of using our tool to evaluate any non established set of guidelines.

As we want the interpretation to be as HTML oriented as possible, the ideal profile for this role is a HF expert with some HTML experience.

Interpretation Structurer

If the guideline interpretation is provided, the structurer provides the corresponding GDL formal structure. He identifies needed HTML elements and provides the formal expression of the interpretation.

The structurer must be experienced with GDL and have very good HTML knowledge. In addition, the generated structure must be review by the interpreter and structurer together to validate it or to modify it if necessary.

Evaluators

The structured set of guidelines can be evaluated at pre publication (by designers) or post publication (by HF experts) of Web sites. In both cases, the evaluator does not need to have any particular GDL knowledge. Our aim is to provide highly customized evaluation reports to meet the needs of both evaluators in term of structure and content of the evaluation report.

10. CONCLUSION

This paper presented an approach for optimising automated evaluation of Web usability guidelines based on the concepts of evaluation sets and conditions. This approach would present some advantages over approaches adopted by existing usability evaluation tools:

- **Targeted guidelines**: it is obvious that traditional evaluation tools cannot evaluate any guideline outside the precompiled set of guidelines hard coded in the evaluation engine of the tool. As for a tool adopting our evaluation approach, its main distinctive feature is its capability to enable the evaluation of any *evaluable* guideline. A guideline is said to be *evaluable* if we can find HTML elements that reflect its semantics and if we can specify the needed evaluation conditions using the vocabularies provided by the evaluation tool. Thus, such a tool should at least be capable of evaluating guidelines that are evaluable by existing tools.
- **Optimisation of the evaluation process**: using the same methodological framework to structure all

guidelines enables us to obtain non conflicting structures: the structuring would show common evaluation sets and common evaluation conditions if exist. By this way, no guideline is evaluated twice. No identical condition but appearing in two evaluation sets or more is checked more than once. We can also combine parsing and evaluation steps to stop evaluating a guideline if one of its evaluation conditions is not verified.

- **High Flexibility in the evaluation process**: separating evaluation process into three independent steps gives many possibilities: we can choose to evaluate a part of a guideline, one guideline or a set of guidelines (possibly from various sources).
- **Identification of conflicts and Similarities among guidelines**: expressing guidelines in a logical and structured form would allow us to easily identify potential conflicts and/or common elements among guidelines.
- **Guidelines Management**: at anytime, a guideline can be added, removed or modified without consequence on the evaluation engine. This independence allows the system to import new sets of guidelines from outside into the tool repository.
- **Customization of evaluation reports**: the high flexibility of our approach should allow us to generate a highly customized evaluation report (e.g., one possible format is given in Fig. 7). In addition to traditional guideline-based evaluation reports generated by existing evaluation tools, we should be able to generate reports based on objects (images, fonts), ergonomic criteria (for usability and accessibility) or any combination of them.

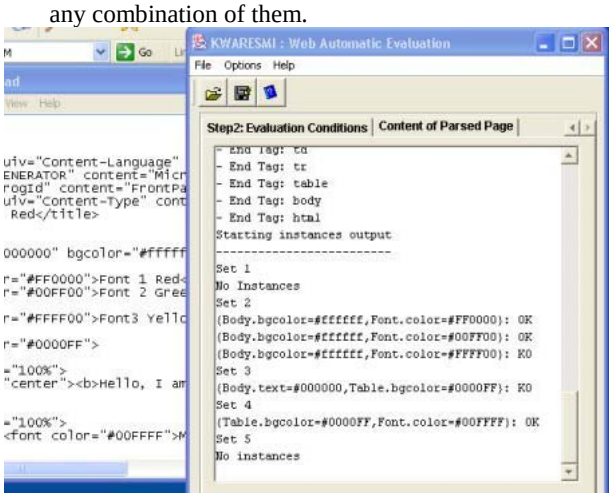


Figure 7. Example of evaluation report.

ACKNOWLEDGEMENTS

We hereby acknowledge the support of the DESTINE research project (http://www.info.fundp.ac.be/~destine) under the umbrella of the WIST Program, Walloon Region and of the SIMILAR network of excellence, the European research task force creating human-machine interfaces similar to human-human communication of the European Sixth Framework Program (FP6-2002-IST1-507609, (http://www .similar.cc).

REFERENCES

1. Abascal, J., Arrue, M., Fajardo I., Garay, N., and Tomas, J., Use of Guidelines to Automatically Verify Web Accessibility, Universal Access in the Information Society, 3(1), 2004, pp. 71-79.
2. A-Prompt: Web Accessibility Verifier, Adaptive Technology Resource Center (University of Toronto) and Trade Center (University of Wisconsin), Canada & USA, 1999. Accessible at http://www.snow.utoronto.ca.
3. Brajnik, G., Automatic Web Usability Evaluation: What Needs to be Done?, Proc. of 6th Conf. on Human Factors and the Web HFWeb'2000 (Austin, 19 June 2000), University of Texas, Austin, 2000.
4. Cooper, M., Evaluating Accessibility and Usability of Web Sites, Proc. of 2nd Int. Conf. on Computer-Aided Design of User Interfaces CADUI'99 (Louvain-la-Neuve, 21-23 October 1999), Kluwer Acad. Publisher, Dordrecht, 1999, pp. 33-42.
5. Farenc, Ch., Liberati, V., and Barthet, M.-F., Automatic Ergonomic Evaluation: What are the Limits?, Proc. of 2nd Int. Workshop on Computer-Aided Design of User Interfaces CADUI'96 (Namur, 5-7 June 1996), Presses Universitaires de Namur, Namur, 1996, pp. 159-170.
6. Farenc, Ch., Palanque, P., Bastide, R., Embedding Ergonomic Rules as Generic Requirements in the Development Process of Interactive Software, Proc. of 7th IFIP Int. Conf. on Human-Computer Interaction INTERACT'99 (Edinburgh, 30 August-3 September 1999), IOS Press, Amsterdam, 1999.
7. Grammenos, D., Akoumianakis, D., and Stephanidis, C., Integrated Support for Working with Guidelines: the Sherlock guideline Management System, Interacting with Computers, 12(3), 2000, pp. 281-311.
8. ISO/WD 9241, Ergonomic Requirements for Office Work with Visual Display Units, International Standard Organization, Geneva, 1992.
9. Ivory, M.Y. and Hearst, M.A., The State of the Art in Automated Usability Evaluation of User Interfaces. ACM Computing Surveys, 33(4), 2001, pp. 470–516.
10. Ivory, M.Y. and Hearst, M.A., Improving Web Site Design, IEEE Internet Computing, Special Issue on Usability and the World Wide Web, 6(2), 2002.
11. Ivory, M.Y., Mankoff, J., and Le, A., Using Automated Tools to Improve Web Site Usage by Users with Diverse Abilities. IT&Society, 1(3), 2003, pp. 195-236.
12. Ivory, M.Y., Automated Web Site Evaluation: Researcher's and Practitioner's Perspectives. Kluwer Academic Publishers, Dordrecht, 2003.
13. Macromedia, Accessibility, Validating Websites for Accessibility Using Macromedia Dreamweaver MX, Macromedia, 2002. Accessible at http://www.macromedia.com/macromedia/accessibility/mx/dw/validation.html
14. Mahajan R. and Shneiderman B., Visual and Textual Consistency Checking Tools for Graphical User Interfaces, IEEE Trans. on Software Engineering, 23(11), 1997, pp. 722-735.
15. Okada, H., Fukuzumi, S., and Asahi, T., GUITESTER2: An Automatic Consistency Evaluation Tool for Graphical User Interfaces Programming Environments, Proc. of 7th IFIP Int. Conf. on Human-Computer Interaction INTERACT'99 (Edinburgh, 30 August-3 Sept. 1999), IOS Press, Amsterdam,

1999, pp. 519-526.
16. Olsina, L. and Rossi, G., Measuring Web Application Quality with WebQEM, IEEE MultiMedia, 9(4), 2002, pp. 20-29.
17. Scapin, D.L., Leulier, C., Vanderdonckt, J., Mariage, C., Bastien, Ch., Farenc, Ch., Palanque, Ph., and Bastide, R., A Framework for Organizing Web Usability Guidelines, Proc. of 6th Conf. on Human Factors and the Web HFWeb'2000 (Austin, 19 June 2000), University of Texas, Austin, 2000.
18. UsableNet, LIFT: Web Preflight and Usability Assistant, UsableNet.com, 2000. Accessible at http://www.usablenet.com.
19. Vanderdonckt, J., Development Milestones towards a Tool for Working with Guidelines, Interacting with Computers, 12(2), December 1999, pp. 81-118.
20. WebCriteria, WebCriteria tool, 2000. Accessible at http://www.webcriteria.com

ADIS: Accessibility and Device Independence System

Regina Bernhaupt
ICT&S Center
University of Salzburg
5020 Salzburg, Austria
Regina.Bernhaupt@sbg.ac.at

Leonhard Brunauer
ICT&S Center
University of Salzburg
5020 Salzburg, Austria
lbrunau@cosy.sbg.ac.at

Manfred Tscheligi
ICT&S Center
University of Salzburg
5020 Salzburg, Austria
Manfred.Tscheligi@sbg.ac.at

ABSTRACT
ADIS (Accessibility and Device Independence System) is a system capable of generating accessible and device independent hypertext structures out of a single data repository. The system helps authors and maintainers of content in hypertext structure to handle accessibility and device independence without much effort. To achieve accessibility of websites ADIS computes an accessible version for every webpage. Device independence is achieved by means of content selection according to the client's capabilities. ADIS additionally delivers an accessible version which can be used on all devices.

ADIS is based on several free available tools and can be used to flexibly generate accessible and/or device independent content. A first version of ADIS is available in June 2005, the system will be finalized in December 2005 – thus ADIS is still work in progress.

Categories and Subject Descriptors
H.5.4 [Hypertext/Hypermedia]

General Terms
Design.

Keywords
Accessibility, device independence, Lenya, Cocoon, mobile devices, interactive TV.

1. INTRODUCTION
ADIS (Accessibility and Device Independence System) is a tool to automatically and dynamically generate accessible and device independent versions of websites. ADIS was designed with three objectives in mind: first, making websites accessible to people with special kinds of visual impairments, second, making websites available for mobile devices and services on interactive TV and third, to generate an accessible version for all supported devices (accessibility and device independence combined). The functionality of ADIS is limited, though. ADIS' applicability lies in web projects focusing on presentation of content – i.e. projects where user interaction is only required to a certain extend. In particular, ADIS should not be considered as a platform for creating web applications.

1.1 Accessibility with ADIS
Making websites accessible to people with special kinds of visual impairments, using for instance a screen reader to gain access, can be addressed in various ways. Following checklists [7], sticking to guidelines [4] and using technical tools [2], [6] can help to improve the accessibility of websites. ADIS' goal is to make it as easy as possible for authors to produce accessible content according to theses recommendations. An emphasis was put on compliance to the Authoring Tool Accessibility Guideline ATAG [11], though some of the guidelines are not met in the current implementation. In particular, the system is intended to support authors in writing proper content, which can in turn be transformed to be accessible.

But making websites – or more generally XML based content – accessible is just the first step. The other major focus of ADIS is to present content device independent. Although this is often realized on the client side – see [3] for an example targeted at mobile devices – ADIS uses a different approach. In ADIS the complete presentation generation is done on the server side. Device independence is achieved by means of content selection according to the client's capabilities.

Additionally, we aim to go one step further and use ADIS not only to make content available on various devices but to be available in an accessible version. For example, the accessible version of a website can be viewed using a PDA. In some cases, of course, using the accessible version can be out of scope. Another focus while designing ADIS' device independence features was put on interactive TV.

1.2 Device Independence
We achieve device independence on the server side by selecting the content according to the client's capabilities. The mechanisms to get these properties are based on the W3C's Composite Capability/Preference Profile CC/PP [9] and the Open-Mobile-Alliance group's User Agent Profile UAProf recommendation. There are several types of properties in both specifications, where ADIS utilizes those related to software and hardware. Hence, user preferences are not taken into account in the current version.

To enable selection of text based content, ADIS asks the authors to provide summaries for longer text sections – e.g. articles. These summaries are selected for devices with small screens or very low bandwidth. This means that the content source for these devices differs from the one delivered to other devices. But since the two versions are stored within the same content source file, authors always see both of them in the editing area, which makes it somehow possible to keep them synchronized.

With the exception of distinct sources for summary and full text sections, all the different output versions are generated out of the same content source, which makes the source format an important issue. One of the requirements is to have a language with

structural tags only. This is mandatory for the content selection process. Since the formatting is done with respect to the generated output version, it is the transformation's job too. Thus, formatting is left out of the content. XHTML2 meets our demands with the additional advantage of being easily transformable to XHTML1.

Currently, modifying content requires the author to know XHTML2, because we use a plain-text editor. To be broadly accepted, a user-friendly editor will be added in June 2005. Furthermore, it should support authors in writing content that conforms to the Web Content Accessibility Guidelines WCAG [4] as stated in ATAG Guideline 3 [11]. Some extensions to this guideline are made to ensure creation of device independent content. For instance to create the above mentioned summary sections.

1.3 Advantages of ADIS

Several projects have been focusing on the development of tools to enhance accessibility or to enable device independence. ENABLED [10] is developing a toolkit to support the transformation of non-accessible web content into accessible forms and multimodal representations. Moreover, the project addresses mobile aspects. On the other side the project IRIS [5] is building a tool to give design recommendations for producing accessible websites. With respect to these projects ADIS can

- produce accessible versions of websites dynamically. For example we do link a so called visual version to its accessible counterpart (see figure 1). The accessible version enhances the website with a glossary, with a special navigation supporting screen readers and it gives the possibility to dynamically change font size (like suggested in [6]).
- produce not only device-independent websites, but additionally deliver an accessible version which is linked on every website. A disabled user in front of a TV screen thus can easily use the accessible version for interactive TV.

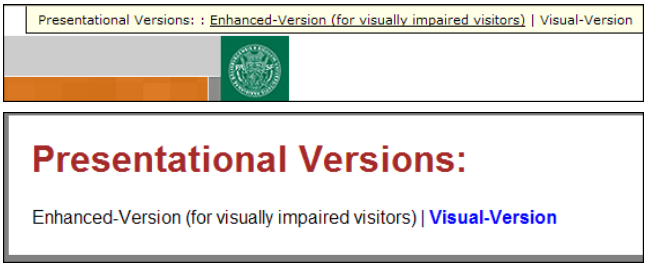


Figure 1: Switching between the standard visual and enhanced accessible version

2. IMPLEMENTATION

ADIS is based on the Apache Lenya CMS (http://lenya.apache.org) and the Cocoon publishing framework (http://cocoon.apache.org). The goal of ADIS is to support accessible and device independent versions of XML based content. ADIS is currently under development and is used in a first version to create a fully accessible version of a website.

To make websites accessible, ADIS must perform a number of transformation steps before content is delivered to the client. XSLT is used to implement these transformations.

2.1 Components and Add-Ons

The Apache Lenya CMS was the perfect solution to build on. Lenya is easily extensible by – what its authors call – 'publication types'. Maybe the greatest advantage of Apache Lenya is that content is not required to be in a single format, like XHTML or HTML. Any valid XML based language can be added to the content resource types, which makes our work a lot easier. Lenya is built upon the Cocoon framework, which provides an XSL transformation pipeline. This is the place where we added our stylesheets. ADIS is an adaptation of Lenya's default publication type with customization of the content resource format, new transformation stylesheets, and modifications in the outline of the Cocoon pipeline.

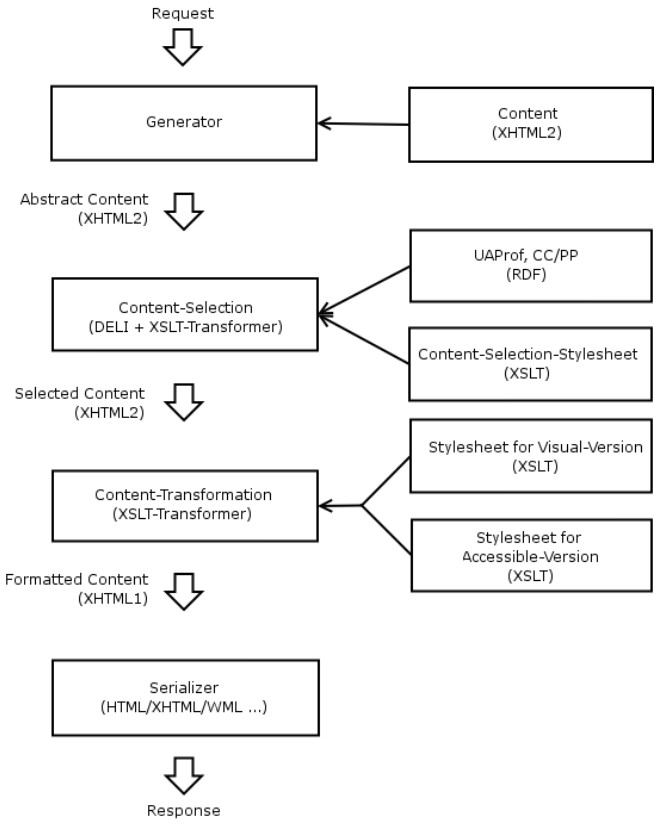


Figure 2: Transformations to generate accessible and device independent content dynamically

The transformation from content to output is laid out as a pipeline (see figure 2). Upon request, a 'generator' reads the appropriate content file and sends SAX events down the pipe. This step can be thought of feeding the pipeline with content. To achieve device independence, the relevant parts of the content source must be selected, which is done in the first transformation step. In the next step another transformation is performed which either creates a visually enhanced or an accessible version. The translation from XHTML2 to XHTML1 is also done in this part of the pipeline.

After this step the transformation process is finished. The transformation to XHTML1 is performed because the pipeline's end – which is called a 'serializer' – needs XHTML1 input. The serializer takes its input – which are still SAX events – and translates it to the desired output format. Its output is in turn delivered to the client.

The stylesheets for the accessibility processing step are based on [8]. In fact there are two distinct transformations, where the client chooses which one is carried out. One of them issues a version with enhanced accessibility, while the other one creates a standard visual version. ADIS analyzes the requested filename and uses the appropriate stylesheet for the filename suffix – e.g. the visual-version stylesheet for `index.vis.html`. A banner at the top of each page enables users to switch between visual and accessible version.

The differences between the two presentational versions are on the one hand structural ones. This includes different navigation mechanisms, glossary, and support of special key-strokes for the accessible version. One the other hand the versions are formatted differently. The accessible version features larger standard fonts, greater contrast, and clear boundaries of structural entities.

As already mentioned, ADIS must select the content that is relevant to the client. This step obviously depends on the client's capabilities like bandwidth, screen size, etc. But the server has to get this information somehow. Fortunately, such a tool has already been integrated with Cocoon. The DELI [1] add-on delivers these capabilities to the transformation process. When a client device is recognized, the properties defined in the appropriate CC/PP or UAProf are available to the content selection stylesheet. Usually, clients provide a link to their profile in the HTTP request. In case they don't, DELI maintains a repository where user-agent-profiles (UAProf files) are stored. This makes it easy to add new devices to ADIS by putting a proper UAProf file to the repository. Then DELI must be told which user agent string (i.e. the string sent along with the HTTP request) should be mapped to which profile. This feature helps ADIS recognize interactive TV or other non CC/PP-aware devices without modifying the client's software.

3. SUMMARY

ADIS is a system enabling users to get accessible versions of websites. Authors are supported in creating accessible websites by reminding them to stick to guidelines. The system is not limited to generate accessible websites, but additionally enables the preparation of content to be displayed on different devices like mobiles or TV screens. However, the most outstanding feature of ADIS is the support of accessible versions of websites on different devices. That is, for example, accessible versions of websites can be "viewed" on TV screens.

4. FUTURE WORK

In June 2005 the first step of ADIS was accomplished providing the functionality to build accessible websites. A focus is set on making ADIS fully satisfying the ATAG 1.0 [11] recommendations, which implies an accessible interface for authors. The next step will be the integration of device independence, which will be finished in September 2005. ADIS will be further enhanced by a special user interface for the CMS and will be tested on several occasions especially within a project on interactive TV. In the next development cycle ADIS will get an accessible and user-friendly interface supporting the user in editing the CSS stylesheets and modifying the transformation stylesheets. The final version of ADIS shall be available in December 2005.

5. REFERENCES

[1] Butler, M.H. DELI: *A delivery context library for CC/PP and UAProf*. External Technical Report HPL-2001-260, 2001.

[2] CAST Bobby. Available at http://www.cast.org/bobby/

[3] Chen, Y., et al. *Detecting Web Page Structure for Adaptive Viewing on Small Form Factor Devices*, WWW 2003, Budapest, 2003.

[4] Chrisholm, W., Vanderheiden, G., and Jacobs, I. *Web Content Accessibility Guidelines 1.0*. 1999. Available at http://www.w3.org/TR/WCAG10/

[5] Darzentas, J. et al. *IRIS: Implementing an Open Environment supporting Inclusive Design of Internet Applications*, Interact03, Zürich, 2003.

[6] Hanson, V. *The User Experience: Designs and Adaptions*, WWW 2004, New York, 2004.

[7] IBM Web Accessibility Checklist. Available at http://www-3.ibm.com/able/guidelines/web/accessweb.html

[8] Katstaller, G. *Accessibility als leitender Faktor bei der Website Entwicklung*. Internal Report, University of Salzburg, Austria, 2005.

[9] Klyne, G., Reynolds, F., Woodrow, C., Ohto, H., Hjelm, J., Butler, M.H., and Tran, L. *Composite Capability/Preference Profiles (CC/PP)*. 2004. Available at http://www.w3.org/TR/2004/REC-CCPP-struct-vocab-20040115

[10] Project ENABLED. Available at http://www.enabledweb.org/

[11] Treviranus, J., McCathieNevile, C., Jacobs, I., and Richards, J. *Authoring Tool Accessibility Guidelines 1.0*. 2000. Available at http://www.w3.org/TR/2000/REC-ATAG10-20000203

Satisfied and Dissatisfied at the Same Time: The 'Must-Have' and 'Attractive' Properties of a User Interface

Timo Jokela
P.O. Box 3000, 90014 Oulu University, Finland
timo.jokela@oulu.fi

ABSTRACT
User satisfaction is one key quality attribute of user interface. In this paper, it is proposed that user satisfaction is not a simple one-dimensional attribute, but two different and independent attributes can be identified: user satisfaction and user dissatisfaction. It is further claimed that different properties of a user interface affect on these two attributes: the absence of 'must-have' properties cause user dissatisfaction and the presence of 'attractive' properties cause user satisfaction. As an implication, it is proposed that all the must-have properties should be identified and included in a user interface, in order to avoid user dissatisfaction. To gain competitive advantage, at least some attractive properties of a user interface should be identified and included in a user interface.

Author Keywords
Usability, user satisfaction, dissatisfaction, quality models

INTRODUCTION
Usability is one important quality attribute of user interfaces. Usability brings many benefits, which include "increased productivity, enhanced quality of work, improved user satisfaction, reductions in support and training costs and improved user satisfaction" [1].

User satisfaction is a key sub attribute of usability. It is included in the different definitions of usability, for example in the ones by Jacob Nielsen [2] and ISO 9241-11 [3]. According to Nielsen, the attributes of usability are learnability, efficiency, remembering, errors, and *satisfaction*. In ISO 9241-11 [3], usability is defined as follows: "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".

User satisfaction is often rated with a Likert-type scale, e.g. 1 = very dissatisfied … 7 = very satisfied. Such a scale implies that a user would be either satisfied or not satisfied, or something in between. In other words, user satisfaction is considered as a single attribute: user satisfaction and user dissatisfaction are regarded as opposite values of one another.

But is user satisfaction really such a simple, one dimensional thing? In this paper, it is proposed that user satisfaction and user dissatisfaction are different attributes, independent of one another. This means, for example, that a user can be satisfied and dissatisfied at the same time. The opposite of satisfaction is not dissatisfaction, but rather, no satisfaction. Similarly, the opposite of dissatisfaction is no dissatisfaction.

Further, it is claimed that the properties of a user interface causing user satisfaction are different from those causing user dissatisfaction. The level of the dissatisfaction is dependent on the presence of *must have* properties while high user satisfaction is achieved with *attractive* properties. As an implication to the design of user interfaces, it is proposed that all the must-have properties should be included in a user interface, in order to avoid user dissatisfaction and poor reputation. To gain competitive advantage in user satisfaction, at least some attractive properties should be included.

The claims of the proposed theory are based on preliminary findings from real life and on two theories from outside the field of HCI. One is the job satisfaction theory of Frederick Herzberg [4]; another one is Noriaki Kano's quality model [5].

In the next section, a set of the author's experiences in real life are given which illustrate the co-existence of user satisfaction and dissatisfaction. In the following section overviews of the theories of Herzberg and Kano are given. Thereafter, a preliminary theory of user satisfaction is outlined where analogies from the theories of Herzberg and Kano are applied to explain the two different attributes of user satisfaction. In the last section, we discuss the implications, limitations, and paths for further research.

USER SATISFACTION AND DISSATISFACTION: EXAMPLES: PERSONAL EXAMPLES FROM REAL LIFE
In the following, I give a set of examples from situations where I have felt satisfaction and dissatisfaction in different ways when using a product (more examples can be found in [6]):

- I bought a coffee maker to our office - a very simple machine. I made the selection mainly based on the fact that the brand was well-know and respected (and expensive) one, which created an expectation that the machine would be more durable than the earlier one. Usability problems became visible after the machine was put into use. People in the office forgot to put the manual switch of the filter in the "open" position and

water flooded out onto the table. Preparing coffee with this device takes many steps because one has to remove and put back many different parts every time.

The usability problems of the coffee maker caused dissatisfaction – even if the device makes good coffee and one might think that the usability problems are not that big. My dissatisfaction in the usability has led me to use it as an example of a product with usability problems in student classes and in presentations.

- A few years back, I had an opportunity to get a smart phone, which combined a mobile phone and a PDA. At first I was excited to combine two devices into one. My use experience, however, soon led to remarkable dissatisfaction. I was used to touch typing with my old PDA. The keypad of the new device, however, was not ergonomically designed for touch typing. I perceived the keypad of the new device as intolerably unusable, and could not stand using the device. Although I very much liked the idea to have only one device, I gave it away after one week. I started to use two separate devices, a mobile phone and a PDA, again.
- I purchased a digital camera and a DVD-player. I am quite happy with these products. This does not mean that the products are free of usability flaws. I cannot use all the properties and I am not totally happy with the use experience. But still I cannot say that I would be very dissatisfied either. I just use the products and don't talk about my use experience to others.

I find that these examples illustrate the following phenomena on user satisfaction:

- A user can be satisfied and dissatisfied with a product at the same time.
- A user can use a product without any specific feelings, even if there clearly are usability problems.
- All the properties of a user interface do not affect the user satisfaction in the same way.
- Sometimes a missing product property may lead to dissatisfaction but in other times not. Sometimes a product property does not really impact on user satisfaction positively but other kinds of product properties seem to lead to high levels of user satisfaction.

OVERVIEW ON HERZBERG'S AND KANO'S THEORIES

In this section, two theories of other fields are presented, Herzberg's theory of job satisfaction, and Kano's quality model.

Herzberg's theory of job satisfaction

To understand employee attitude and motivation, Frederick Herzberg carried out studies to find out which factors in an employee's work environment caused satisfaction and dissatisfaction [4]. His studies included interviews in which employees were asked what pleased and displeased them about their work. Herzberg found that the factors causing job satisfaction were different from those causing job dissatisfaction. He developed a theory to explain these results. He identified job satisfaction and job dissatisfaction as separate attributes. Factors leading to dissatisfaction include for example company policy, work conditions and salary; factors leading to satisfaction are such as achievement, recognition and responsibility, Table 1. .

Dissatisfiers	Satisfiers
Company and administrative policies	Work itself
Supervision	Achievement
Salary	Recognition
Interpersonal relations	Responsibility
Working conditions	Advancement

Table 1. Herzberg's theory: different factors cause job dissatisfaction and job satisfaction

Because factors causing satisfaction are different from those causing dissatisfaction, Herzberg concluded that the two feelings cannot be treated as opposites. The opposite of satisfaction is not dissatisfaction, but rather, no satisfaction. Similarly, the opposite of dissatisfaction is no dissatisfaction.

Kano's quality model

Noriaki Kano's quality model [5] was developed in Japan in 1980's. It is a general quality model, which explains the relationship between the different product properties and customer satisfaction. Must-have properties represent the quality that customers expect from a product. If the must-have properties are not present in the product, the consequence is customer dissatisfaction. The more-is-better properties have a linear impact on customer satisfaction. The attractive properties make a product stand out from the others and provide high customer satisfaction. These properties address unspoken or unexpected needs of the customer that when satisfied, create a positive surprise, and lead to high-levels of satisfaction.

The author has applied Kano's model to usability [7], defining the concepts must-have, more-is-better, and attractive usability, Figure 1. The lower curve of the model reflects must-have usability that the customer expects from a product. The absence of must-have usability will lead to customer dissatisfaction but meeting the must-have usability it is not enough for attaining customer satisfaction. The 'best one can get' with must-have usability features is avoidance of user dissatisfaction.

Increased customer satisfaction can be achieved through more-is-better usability which has a linear impact on customer satisfaction. However, to achieve dramatic impact on the satisfaction of customers attractive usability is required.

In Figure 1, the positions '1' on the blue and red curve mean a good setting: a user is both satisfied and not dissatisfied. The positions '2', on the other hand, imply a poor situation: not satisfied and dissatisfied. But the setting also could be e.g. '1' on the blue curve and '2' on the red curve, meaning a situation when the users has both very positive and negative feelings towards using the product.

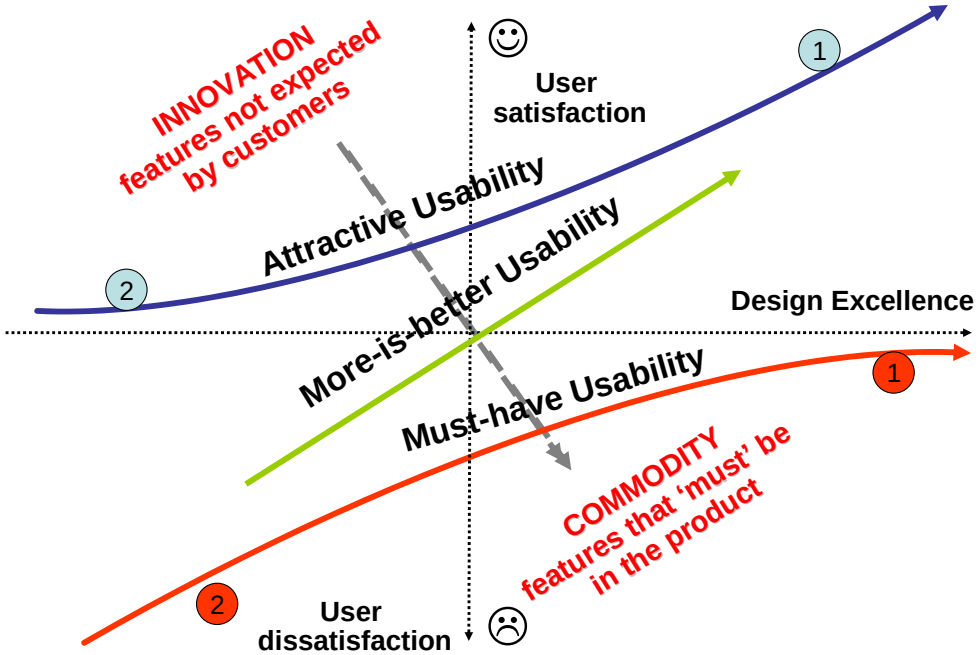


Figure 1: Must-have, more-is-better and attractive usability

Kano's model further claims that attractive properties become must-have ones in the course of time. Properties that were exciting and exceeded customers' expectations at one time become part of quality that a customer cannot live without. When I got my first PDA, I was totally happy with the 'thumb typing' keyboard. The possibility of touch typing of the newer PDA model was attractive quality to me. Today, it represents must-have quality – I did not accept a new PDA device without possibility for touch typing.

Avoiding use dissatisfaction is critically important because it leads to bad reputation [7]. My dissatisfaction with the user interface of the coffee maker led me to use it as an example of a product with usability problems in student classes and in presentations, probably damaging bit of the reputation of the product. What may be of concern to the manufacturer of the smart phone is that I have frequently told others about my dissatisfaction on using the product.

A PRELIMINARY THEORY OF USER SATISFACTION

The real-life examples (section 2) indicate that the concept of 'user satisfaction' is not a simple thing. In those specific situations, the attribute 'user satisfaction' alone is not adequate to describe the user's state: the state just could not be rated with one Likert-type of scale. Instead, it is logical to think that a user was satisfied and dissatisfied at the same time.

In this paper it is claimed that the experiences on user satisfaction from real life could be explained through analogies from the theories of Herzberg and Kano. One could think that two different attributes, user satisfaction and user dissatisfaction would exist, analogous with the attributes job satisfaction and job dissatisfaction of Herzberg's theory. User satisfaction and user dissatisfaction would be independent attributes, as illustrated in Table 2.

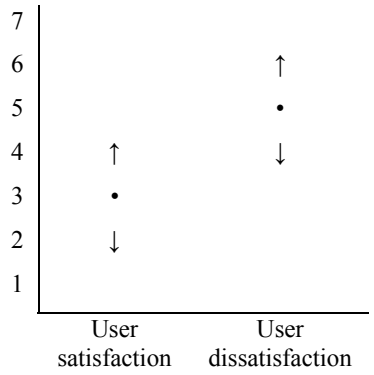


Table 2. Levels of user satisfaction and user dissatisfaction are independent of each other and can be rated separately e.g. using a Likert-type of scale

What properties could cause user satisfaction or dissatisfaction then? Herzberg identified different factors that cause job satisfaction and job dissatisfaction. These factors are, however, domain specific (salary, working conditions, recognition, responsibility, etc.), and cannot be applied as properties of a user interfaces, affecting user dissatisfaction and user satisfaction.

The Kano model provides a sense-making alternative for defining these properties. The must-have properties are features that cause user dissatisfaction, and the attractive properties cause user satisfaction, Table 3.

Dissatisfiers	Satisfiers
Must-have usability properties	Attractive usability properties

Table 3. Different properties affect on user dissatisfaction and user satisfaction

Even apparently 'big' problems may be acceptable to the users if the must-have expectations are met. On the other hand, seemingly minor problems in a user interface may lead to user dissatisfaction if the user feels that the must-have quality is not met.

Independent from the existence of dissatisfiers, the existence of attractive usability properties provides user satisfaction.

In the examples of the digital camera and DVD player, the devices were first of the kind that I had. I had no prior experience on using those kinds of products, and I did not have specific expectations of their usability. Although these products had usability problems, my use experience did not lead to dissatisfaction because the products met my must-have usability expectations.

The situation was different in the cases of the coffee maker and the smart phone. I had earlier experience on both kinds of products. I was used to 'normal' simple coffee makers, with automatic switch in the filter and with only few removable parts. This experience set the level to my must-have usability expectations which were not filled when using the new one. My must-have expectation with the smart phone was to be able to touch type with the keyboard, and this need was not fulfilled. In both of these cases I was dissatisfied because the devices did not meet my must-have expectations.

My first PDA had a 'thumb typing' keyboard. The keyboard designed for touch typing of my second PDA represented attractive usability to me – I really wanted to have it after seeing other people touch typing with the device. Examples of attractive quality can be continuously found in the new cars.

In summary, the claim of this paper is: *Two different and independent attributes related to user satisfaction can be identified: user satisfaction and user dissatisfaction. Different product properties affect on these two attributes: so called must-have properties cause user dissatisfaction, and so called attractive properties cause user satisfaction.*

In the following, I try to formalise this theory using Robin Dubin's framework for defining a theory [8]. Many of the statements below are analogous with the statements of Herzberg in the job satisfaction theory.

Units

Must-have properties: those useful properties that a user expects the product to have.

Attractive properties: those useful properties that a user does not expect the product to have

Satisfaction

Dissatisfaction

Laws of interaction

Inverse relation between the level of dissatisfaction and the existence of the must-have properties in the product

Positive relation between the level of satisfaction and the existence of the attractive properties in the product

Boundaries

When a user gets a new product for use

When a user has options for alternative products

When the user is knowledgeable of other products

Limiting values

Dissatisfaction: 0 … the user denies using the product

Satisfaction: 0 … high but unspecified

System states

State of equal dissatisfaction and satisfaction

Satisfaction level far higher than dissatisfaction

Dissatisfaction level far higher than satisfaction

Proposition (truth statements of the model)

Individual's attitudinal orientation toward using a product is the sum of a level of satisfaction and a level of dissatisfaction

An individual may feel no satisfaction and no dissatisfaction about using a product – he or she may be genuinely indifferent

The level of satisfaction and the level of dissatisfaction towards using a product are independent of each other

DISCUSSION

User satisfaction is a key attribute of usability; therefore, it is important to understand what exactly user satisfaction is about. Traditionally, user satisfaction has been regarded as one-dimensional attribute of usability. In this paper, it is proposed that user satisfaction is not a simple one-dimensional thing, but two different and independent attributes can be identified: user satisfaction and user dissatisfaction. A user can, for example, be satisfied and dissatisfied at the same time.

It is further claimed that different properties of a user interface affect on these two attributes.

- The absence of must-have properties causes user dissatisfaction but the presence of must-have properties does not lead to user satisfaction.
- The presence of attractive properties leads to user satisfaction but the absence of attractive properties does not lead to user dissatisfaction.

The theory presented in this paper could be useful when user satisfaction is considered as a relevant quality attribute of a user interface (which probably often is the case…). The theory would imply the following considerations in the design of a user interface:

- It is important that all the must-have usability properties are identified and included in the quality model of a user interface. The must-have properties need to be present in a user interface in order to avoid user dissatisfaction and thereby poor product reputation in the market place.
- Attractive properties may or may not be included in the quality model. If no attractive properties are included, the result would be 'no specific feelings': not satisfied and not dissatisfied (presuming that must-have properties are present). This could be in many cases an acceptable goal in user interface design: users would not be dissatisfied but the use interface would be not a competitive property.
- To gain competitive advantage in user satisfaction, at least some attractive properties leading to user satisfaction should be included in the quality model. Introducing attractive usability properties in a user interface would be a means to differentiate in the market place and gain competitive advantage.

Limitations

The claims of this paper should be understood as preliminary ones. On one hand, the empirical evidence is only very preliminary: a limited number of qualitative observations on personal behaviour (although the examples are from in real life). On the other hand, the theory is deduced from theories from other fields; such theories do not necessarily work analogously in the context of user satisfaction. Further, the exact statements in the theory definition should not be understood as definite ones. In the formulation of the statements, we used Dubin's framework for defining a theory, and analogies from the statements from Herzberg's theory. The formalisations of the statements most probably need to be refined.

Research topics

If anything, the things presented in this paper leave space for further research.

Systematic user studies are needed for refining the claims of the theory. Would further studies confirm the hypothesis of the validity of the two attributes (user satisfaction, user dissatisfaction), their mutual independence, and the user interfaces properties causing them? An open issue, among others, is the impact of the 'use life-cycle' on the user satisfaction. What would be the exact boundaries of the theory? Would a user be annoyed on the dissatisfiers only in when starting to use a product but get used to the product in the course of time? Would user dissatisfaction be especially strong immediately after a user has purchased and (or) started to use a product? Would the dissatisfaction be reduced over time? Later, hypothesis statements for empirical validation should be formulated, and their validity empirical justified.

Another research question is: how to identify the must-have and attractive properties? Methodological guidance should be developed for identifying both kinds of properties. Probably different methods are required for these two purposes.

ACKNOWLEDGEMENTS

I thank Nomadic Media – a European ITEA project – for providing an environment where the ideas of this paper were generated.

REFERENCES

1. ISO/IEC, *13407 Human-Centred Design Processes for Interactive Systems*. 1999: ISO/IEC 13407: 1999 (E).
2. Nielsen, J., *Usability Engineering*. 1993, San Diego: Academic Press, Inc. 358.
3. ISO/IEC, *9241-11 Ergonomic requirements for office work with visual display terminals (VDT)s - Part 11 Guidance on usability*. 1998: ISO/IEC 9241-11: 1998 (E).
4. Herzberg, F., *The motivation to work*. 2d ed. 1959, New York,: Wiley. 157 p.
5. Kano, N., N. Seraku, F. Takahashi, and S.-i. Tsuji, *Attractive quality and must-be quality.* The Journal of the Japanese Society for Quality Control, 1984. **14**(2): p. 39 - 48.
6. Jokela, T., K.-P. Aikio, and I. Jounila. *Satisfied and dissatisfied at the same time. A preliminary two-factor theory of user satisfaction*. in *UACHI 2005*. 2005. Las Vegas. Invited.

7. Jokela, T., *When Good Things Happen to Bad Products: Where are the Benefits of Usability in the Consumer Appliance Market?* ACM interactions, 2004. **XI.6**(November + December): p. 28-35.

8. Dubin, R., *Theory building*. 1969, New York: Free Press. ix, 298p.

Case Studies on a Quality Model based on the ISO 9241-11 Definition of Usability

Timo Jokela
P.O. Box 3000, 90014 Oulu University, Finland
timo.jokela@oulu.fi

ABSTRACT
We explored a quality model based on the ISO 9241-11 definition of usability - "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 determining usability requirements in industrial development projects. Overall, working with the quality model was perceived useful and sense-making both by participants. On the other hand, some concepts of the definition were perceived found unclear, and producing usability requirements and managing the process were challenging.

Author Keywords
Usability, ISO 9241, quality models

INTRODUCTION
Usability is one of the most important quality attributes. Usability brings many benefits, which include "increased productivity, enhanced quality of work, improved user satisfaction, reductions in support and training costs and improved user satisfaction" [1].

Usability has not been defined consistently, and various definitions exist. Probably the best-know definition of usability is by Jacob Nielsen [2]: usability is about learnability, efficiency, memorability, errors, and satisfaction. ISO 9126 defines usability as "a set of attributes that bear on the effort needed for use, and on the individual assessment of such use, by a stated or implied set of users".

The definition of usability from ISO 9241-11 [3] - *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* – has probably become the main reference of usability. In addition that it is largely recognized in recent literature, the new Common Industry Format, CIF, for usability testing [4] - supported by a number of corporations and other stakeholders - uses this standard definition as the reference of usability.

The call for the workshop states, "a quality model must be built at the outset" of a development project. This applies to usability as well: *usability requirements* should be determined in the beginning of a development project.

There is a growing interest on the determination of usability requirements using the ISO 9241-11 definition. For example, the ongoing activity to create the 'Common Industry Format for Usability Requirements', CIF-R has already led to a preliminary version [5] where the requirements format is based on the ISO 9241-11 definition.

The standard ISO 9241-11 and the CIF-R provide formats and examples for how to determine usability requirements using the definition of usability of ISO 9241-11. They mainly describe and explore the concepts and formats of the requirements document and propose measures (such as error count) for the definition of usability requirements. Research carried out in European projects has produced guidance and templates for describing users, tasks, and environments of use [6] and [7]. Most of other usability engineering literature such as [2], [8], [9], [10], [11] and [12] do not explicitly use the ISO 9241-11 definition.

But is the definition of usability of ISO 9241-11 a useful one in defining usability requirements? In spite of the popularity of the ISO 9241-11 definition, only a limited number of empirical studies have been reported on the practical use of the definition. One of the few is by Bevan & al [13] who used the definition in usability testing. They, however, only focus on usability evaluation, and do not discuss the definition of usability or the production of the usability requirements.

In this paper, we report our findings on using a usability quality model based on ISO 9241-11 definition of usability in determining usability requirements in different product development projects. Our research approach is qualitative and interpretative. The conclusions presented in this paper are our interpretations from our case studies, based on qualitative data from observations at usability requirements determination workshops and from feedback from the participants in those workshops.

USABILITY QUALITY MODEL
The usability quality model - our interpretation of the ISO 9241-11 definition (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) - is presented in Figure 1.

When defining the quality model, we basically followed the quality model framework of Firesmith [14], ending up with a hierarchical tree structure. We found it reasonable to two levels of subfactors ('subfactor', 'sub-subfactor'), in order to make the model clear. If the

high-level quality factor is 'usability', we identify the following levels in the quality model:

- The subquality factors are 'Usability for users 1'; 'Usability for users 2'; etc.
- The sub-subquality factors are 'Usability for task 1'; 'Usability for task 2'; etc.
- The quality criteria for each sub-subfactor can be selected from 'Effectiveness', 'Efficiency' and 'Satisfaction'
- For each criteria, different measures can be identified
- The minimum requirement for each individual criterion is defined.

CASE STUDIES

Altogether, we have explored the quality model in determining usability requirements in eleven case studies so far. The cases have been development projects of different kinds of applications: mobile services, telecommunication software, and transportation and healthcare equipment. The sizes of the companies varied from multi-national to SME ones.

Our practical implementation for determining the usability requirements has been workshops, or 'stakeholder meetings' [6] where the participants are the key stakeholders of a development project and usability experts (in our case at the same time researchers) act as facilitators of the meeting. In the meetings, the idea has been to systematically elicit and analyze the knowledge that the participants have on users and formalize it into usability requirements.

Such workshops can be regarded as the first iteration in the requirements determination cycle. We did not carry out, for example, extensive field studies to understand the user requirements. In other words, our goal was not to determine the absolutely 'right', final usability requirements. Rather, in most cases the workshops were kick-off meetings – often with training objective - for usability work in a starting development project.

In the requirements process, we tried to follow the quality model (Figure 1) as follows:

- We first identified the users
- For each set of users, we then determined the user tasks, through identifying the user goals, and the context of the tasks
- We then determined appropriate quality criteria (effectiveness, efficiency and/or satisfaction)
- We finally defined the minimum usability requirements

In practice, the requirements process was not straightforward: iteration and prioritizing took place at various phases of the process.

There was variation in the cases in the extent to which the usability requirements were determined. In some cases, only a relatively small part of the requirements could be determined; in other cases the requirements were determined more extensively. We were not able to carry out the requirements process fully except for one rather simple case. Some cases composed of several stakeholder meetings while other cases were carried out in one session. The variation was partially due to the different times and resources available for the stakeholder meetings, partially due to the methodological challenges that we met.

On identification and categorization of users

The first phase of the requirements process, the identification of users was not very easy but not very difficult either (compared with some of the latter steps).

We ended up in most cases to use the job role as the main criterion for categorisation rather than e.g. the experience of users - novice, expert, etc. - or cultural factors (for different strategies for categorizing users: ref. e.g. to [3]). With this strategy we achieved consistency with the next phase: users within each category would share the same goals.

On identifying user goals and user tasks

Every measure of usability is a function of users achieving their goals. We therefore regarded the identification of user goals as a very critical aspect in the usability requirements process.

The existing methods did not provide very concrete guidance for how to identify user goals. For example, a definition of user goal is provided in ISO 9241-11 ('intended outcome) and the difference between 'goal' and 'task' is recognized but this distinction is not really followed. The standard states that usability measures can be specified "for overall goals (e.g. produce a letter) or for narrower goals (e.g. perform search and replace)". However, "produce a letter" and "perform search and replace" are clearly tasks rather than goals.

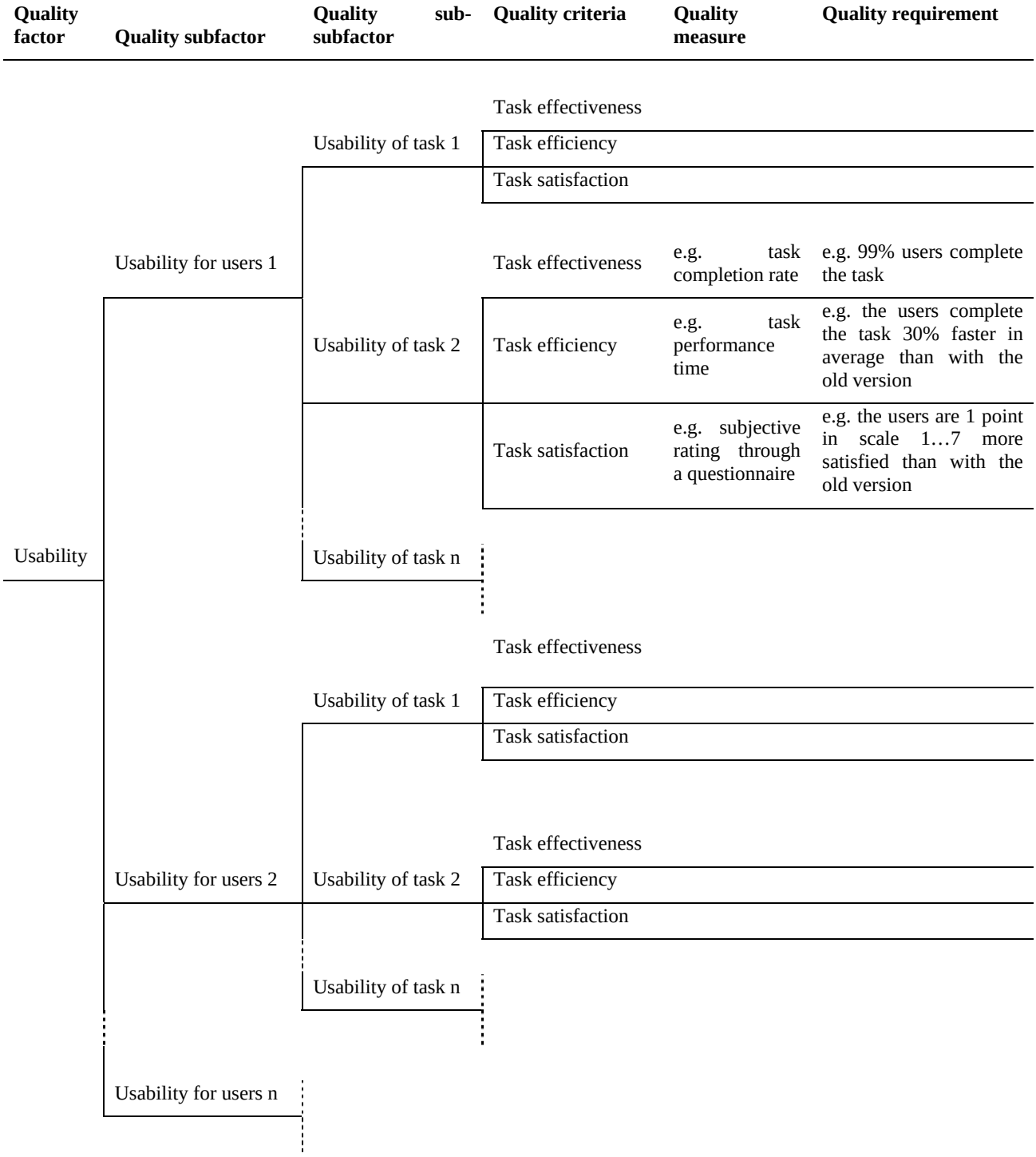


Figure 1. The usability quality model derived from the ISO 9241-11 definition of usability

Probably the most concrete advice in the literature is to select the most important user goals: "focusing … on the most important user goals may mean ignoring many functions, but is likely to be he most practical approach" [3]. General guidance is also given, such as "the users' overall goals should be studied" and "a typical outcome of a task analysis is a list of all the things users want to accomplish with the system" [2]. - CIF-R [5] describes only the format of usability requirements document but

does not provide guidelines for how to generate the contents of the requirements.

In summary, we had to make an interpretation of our own on what exactly means a 'user goal'. We interpreted a goal strictly as something to be accomplished; i.e. not as a task. For example, in one case the participants proposed "final testing" as being a user goal. We discussed the nature of the statement and concluded that it is a task rather than a goal. After a bit of brainstorming, we formalized the statement into the format of goal: "to have the fault-free products separated from the faulty ones", Figure 2.

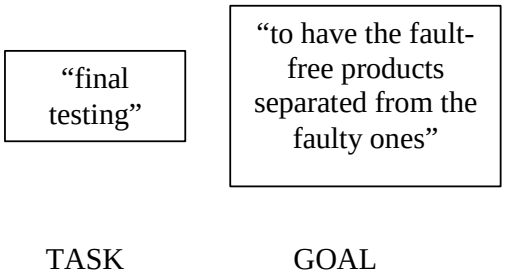


Figure 2. The difference between a task and a goal

The distinction between 'goal' and 'task' is useful because goals are the ultimate things one wants to accomplish with a product; tasks are only means for achieving the goals. In some cases, for example, this distinction helped the project staff to realise that reaching a goal could be automated, without any user task.

Selecting quality criteria and determining usability requirements

Another challenging phase of the process was to determine measurable target levels for the usability attributes (effectiveness, efficiency, satisfaction).

We found the existing guidance too general here, too. Basic guidance for the measures of effectiveness, efficiency and satisfaction is provided in [3] and [4]. The standard ISO 13407 [1] identifies a process 'Specify the user and organizational requirements', including statements such as "provide measurable criteria against which the emerging design can be tested", and that the requirements should "be stated in terms that permit subsequent testing and should be confirmed or updated during the life of the project". The experience report by Bevan & al [15] guides: "For each chosen task and user type, estimate…" and "usability for the new system should be at least as good as for the old system". The handbook of Maguire [6] provides some guidance for determining qualitative usability goals but very little for quantitative ones. Some methodologies, e.g. [9], do not address this issue at all.

In determining the target levels of effectiveness, efficiency and satisfaction, we typically ended up using relative measures: how much better the new product should be compared with an earlier version. For example, a target could be that user satisfaction should be one point better in scale 1…7 compared with an earlier product (vs. an absolute measure, such as user satisfaction should be '6').

Generally, however, we could not reach a very systematic strategy for carrying out this phase. For example, setting the target level for user satisfaction was more or less just 'guessing'.

CONCLUSIONS

We first draw conclusions on the different concepts of the definition of usability, and then on producing the usability quality model.

Clearer interpretations of some concepts are needed

For sensible requirements determination, a clear and shared understanding of concepts is a prerequisite. More precise definitions of some concepts of the ISO 9241-11 definition are needed:

- User goal. The concept 'user goal' is not clearly defined in the standard. For example, is "carry out the final testing" (in production line) a user goal? Or should it be "have the faulty products separated from the ones to be delivered to the customer". 'User goal' is such a central part of the definition that a conceptually clear interpretation is required. As said, we interpreted goal as something to be accomplished. It should describe a state rather than 'doing something'.

  Working with user goals is useful. Still, we ended up with using task names rather than goals in the quality model (Figure 1). The names of tasks just are simpler and easier to discuss than the names of the goals (Figure 2).

- Context of use. The term 'context of use' not quite consistently used in the definition of usability. It refers to users and tasks, which means redundancy in the definition: 'user' is mentioned in the definition also explicitly. We did not have true problems with the definition. Still, we find more consistent to use the term 'environment of use' rather than 'context of use'.

- Satisfaction. Satisfaction is a bit vague concept. A common question was, is satisfaction a function of effectiveness and efficiency? When it is exactly relevant to define requirements in terms of satisfaction? The concept satisfaction should probably be elaborated. One idea is to understand user satisfaction and dissatisfaction as separate and independent attributes [16].

ISO 9241-11 driven requirements process is a useful thinking process

Our conclusion is that the process of producing usability requirements based on ISO 9241-11 definition of usability is useful, sensible and effective training of usability. The participants, software designers and other stakeholders of project teams gave comments such as "a new and meaningful way of thinking" and "we definitely should have done this in earlier projects". They further reported that the process "opened their eyes" and the results represent a "totally new and meaningful perspective to product requirements". We find as an example on the perceived usefulness of the definition a

case with three half-day sessions. Two product managers from the marketing department – typically very busy people - actively participated in all the sessions. The number of participants even increased in the last session, after the word about the meetings had spread within the company.

We find that this usefulness lies in the substance of the definition; the definition is useful because of its contents, not because it has 'the authority' of a standard. We did not even consider taking any other definition as the basis of our work throughout the cases.

Creation of the usability quality model and usability requirements is challenging

Forming a quality model and usability requirements based on the ISO 9241-11 definition – determining users, user goals, effectiveness, efficiency, satisfaction, context of use – is not easy.

- Although 'user' is a clear concept, it is not easy to determine an appropriate set of user categories.
- Identification of user goals is even more difficult. The conceptualization of what is 'user goal' is not easy; it is quite difficult to make a distinction between a task and a goal. Another difficulty is in the identification of the user goals: how does one make sure that all the important goals are identified?
- The usability requirements, i.e. the levels of effectiveness, efficiency, and satisfaction are difficult to determine. As such, it is not very difficult to formalize measurable requirements, such as "90% of users should complete a task correctly". But the critical question is: Does the requirement "90% of users should complete a task correctly" truly depict a usable product?
- Overall, the determination of usability quality model using the ISO 9241-11 definition is a complex task to manage. In typical cases, many different user groups can be identified. Each user group may have many different goals that also may vary among the different user groups. The levels of goals may be different in terms of effectiveness, efficiency, and satisfaction – also in cases where different users share the same goals (Figure 1). The facilitators had to continuously consider how to manage the complexity: which the issues to be worked with and which ones to postpone or omit. Especially with systems with many different user groups and a large number of tasks, one had to tackle with 'space explosion' all the time (the number of items, and their combinations became so large).

DISCUSSION

We explored the definition of usability from ISO 9241-11 in defining the usability quality model and determining usability requirements. We found the definition useful and sense-making but all the concepts are not clearly described. The case studies revealed the complexity of the requirements process when using the definition.

The definition of usability of ISO 9241-11 is a popular and important reference of usability but very little experience reports exist. Thereby, we find that our empirical research results on using the definition in practice are relevant.

Our results can be regarded as preliminary ones. The case studies were carried out in real contexts but were not very extensive: we were able to carry out the requirements process only partially, and could not follow up the impact of the requirements in the product designs. Another limitation is types of application and products that we had: they were business-to-business or other 'useful' applications. They did not cover any personal 'fun' or web applications. Different cases might have brought other kinds of viewpoints (such as fun or pleasure) to the definition. One may also regard the qualitative nature of the research: the results are our interpretations from our observations and the qualitative feedback gathered from the participants of the workshops. The feedback was gathered in most cases with a simple questionnaire ("Give 3 pros and 3 cons"). In depth interviews might have revealed other kinds of findings.

The study has raised some questions by some audience:

- What are the weaknesses of ISO 9241-11 definition e.g. when defining learnability requirements?
- How does the approach relate to the claim that it is impossible to design systems for many different user groups that are actually usable [17]?

On the first question: In our cases studies there really was no specific emphasis on the learnability aspect, and the research does not address this question. Learnability, however, was sometimes naturally considered when determining the target levels for the usability criteria.

On the second question: The idea is to determine the target levels of usability criteria in our cases. It does not imply that all target levels would present high usability. Basically, some target levels could also be quite low. In other words, setting usability target levels is a totally different issue as claiming that a product could be usable to all.

Implications to practice

The results of this study encourage using the definition of ISO 9241-11 in development projects. Determining usability requirements as teamwork with a project team using the definition is very effective training on usability. It helps in identifying usability driven design drivers to products under development. If it is not feasible to assign resources for several stakeholder meetings, even one day workshop could be useful. Anyway, the cases should be real, i.e. development projects that is about to start.

We used prioritizing - e.g. focused on the most important user groups only. Probably this is needed in other cases, too. It may also be practically necessary to skip some phases of the process (in most cases, we skipped the detailed analysis of the environments of use).

Further research topics

More methodological guidance should be developed especially for determining user goals, determining target

levels of usability, and generally managing the complexity of usability requirements process. Further – although we found the definition of usability of ISO 9241-11 at a general level sensible – its concepts should be clarified and defined more consistently.

There is a need for 'full' case studies where usability requirements could be explored 'to the end' and one could be able to follow the impact of the requirements throughout the project life-cycle. This kind of study would help in finding solutions to the management of complexity of the requirements, as well as finding effective solutions to other challenges of the requirements process, such as determining the 'right' target levels of effectiveness, efficiency, and satisfaction.

REFERENCES

1. ISO/IEC, *13407 Human-Centred Design Processes for Interactive Systems*. 1999: ISO/IEC 13407: 1999 (E).
2. Nielsen, J., *Usability Engineering*. 1993, San Diego: Academic Press, Inc. 358.
3. ISO/IEC, *9241-11 Ergonomic requirements for office work with visual display terminals (VDT)s - Part 11 Guidance on usability*. 1998: ISO/IEC 9241-11: 1998 (E).
4. ANSI, *Common Industry Format for Usability Test Reports.*. 2001: NCITS 354-2001.
5. NIST, *Proposed Industry Format for Usability Requirements. Draft version 0.62. 8-Aug-04*. 2004.
6. Maguire, M., *RESPECT User-centred Requirements Handbook. Version 3.3*. 1998, HUSAT Research Institute (now the Ergonomics and Saftety Research Institute, ESRI), Loughborough University, UK.
7. Thomas, C. and N. Bevan, *Usability Context Analysis: A Practical Guide. Version 4.04*. 1996, National Physical Laboratory: Teddington.
8. Wixon, D. and C. Wilson, *The Usability Engineering Framework for Product Design and Evaluation*, in *Handbook of Human-Computer Interaction*, M. Helander, T. Landauer, and P. Prabhu, Editors. 1997, Elsevier Science B.V: Amsterdam. p. 653-688.
9. Beyer, H. and K. Holtzblatt, *Contextual Design: Defining Customer-Centered Systems.* 1998, San Francisco: Morgan Kaufmann Publishers. 472.
10. Hackos, J.T. and J.C. Redish, *User and Task Analysis for Interface Design*. 1998: Wiley Computer Publishing.
11. Mayhew, D.J., *The Usability Engineering Lifecycle*. 1999, San Fancisco: Morgan Kaufman.
12. Rosson, M.B. and J.M. Carroll, *Usability Engineering. Scenario-Based Development of Human-Computer Interaction*. 2002: Morgan Kaufmann Publishers.
13. Bevan, N., N. Claridge, M. Maguire, and M. Athousaki. *Specifying and evaluating usability requirements using the Common Industry Format: Four case studies*. in *IFIP 17th World Computer Conference 2002 - TC 13 Stream on Usability: Gaining a Competitive Edge*. 2002. Montreal, Canada: Kluwer Academic Publishers.
14. Firesmith, D., *Using quality models to engineer quality requirements.* Journal of Object Technology, 2003. **2**(5): p. 67-75.
15. Bevan, N., N. Claridge, M. Athousaki, M. Maguire, T. Catarci, G. Matarazzo, and G. Raiss, *Guide to specifying and evaluating usability as part of a contract, version1.0. PRUE project*. 2002, Serco Usability Services: London. p. 47.
16. Jokela, T., K.-P. Aikio, and I. Jounila. *Satisfied and dissatisfied at the same time. A preliminary two-factor theory of user satisfaction*. in *UACHI 2005*. 2005. Las Vegas. Invited.
17. Cooper, A. and P. Saffo, *The Inmates Are Running the Asylum: Why High Tech Products Drive Us Crazy and How To Restore The Sanity*. 1999: Sams. 261.

A Quality Model for Early Usability Evaluation

Silvia Abrahão **Oscar Pastor**
Department of Information Systems and Computation
Valencia University of Technology
Camino de Vera s/n, 46071 Valencia, Spain
sabrahao, opastor@dsic.upv.es
+34 96 387 7000

Luis Olsina
GIDIS, Department of Informatics
Engineering School at UNLPam
Calle 9 y 110 (6360) La Pampa, Argentina
olsinal@ing.unlpam.edu.ar
+54 2302 430207, Ext. 6501

ABSTRACT
Due to the increasing interest in the Model Driven Architecture (MDA) paradigm, the conceptual models have become the backbone of the software development process. However, very few works have been proposed for the usability evaluation at this level. The means for ensuring usability as part of the software modeling process are needed. A framework that improves final software system usability by modeling usability aspects is introduced in this work. This framework has been defined in the context of the OlivaNova Model Execution technology, the industrial implementation of an automatic software production method called OO-Method. Specifically, we present a quality model that is the basis for the usability framework. The usability of a software system is evaluated and improved at the conceptual model level using this model. It focuses on the correspondences between the abstract definition of the user interface in terms of presentation patterns (Problem Space) and its specific software implementations (Solution Space).

Keywords
User Interfaces, Conceptual Models, Quality Models, Model-Driven Architecture.

ACM Classification Keywords
Information interfaces and presentation: Miscellaneous.

INTRODUCTION

Many approaches to evaluate the usability of software systems have been proposed in the literature [4] [9] [14] [22] [29] [30] [33]. Most of these proposals focus on defining a set of attributes that explains usability and on developing guidelines and heuristics for testing usability. Several techniques such as usability testing [22], and usability inspection [23] can be used to evaluate the usability of systems. However, none of these techniques focus on the artifacts used in the early stages of system development. In addition, some proposals do not explicitly define a quality model that explains the relationships among the different usability attributes.

Other studies have demonstrated that 50% of the implementation stage time is dedicated to user interface construction [21]. This has motivated the development of tools for the automatic generation of user interfaces. Among these tools, those that follow the Model-Driven Architecture (MDA) approach [18] seem to be appropriate for the development of user interfaces. These approaches take a requirements specification that is converted into different conceptual models as input. These models are used to automatically generate a user interface that is compliant with the captured requirements. In this context, tools should help to create usable user interfaces; unfortunately, almost no tools have provided explicit support for usability evaluation.

In this paper, we introduce a framework for usability evaluation based on conceptual models. This framework has been defined in the context of the OlivaNova Model Execution (ONME) technology [24], which is the implementation of an automatic software production method called OO-Method [27]. According to Figure 1, the ONME development process starts with the construction of a conceptual model from a set of functional requirements. The usability of a software application can then be evaluated at the Presentation Model [20] level. This model describes the abstract user interface using a set of presentation patterns. The objective of the evaluation is to provide feedback to the analysts for changes in order to avoid usability problems in the generated application. This iterative process combines model development and usability evaluation and can be applied until the Presentation Model has the required level of usability.

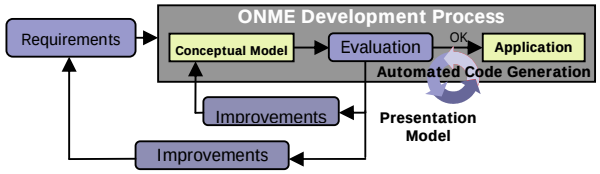


Figure 1. ONME development process and usability evaluation

Note that the evaluation is made *without* any user testing. This allows very early usability evaluation by verifying the conceptual model properties that are related to usability. Furthermore, we are not only interested in identifying possible usability problems at the Presentation Model level, but we are also interested in the way in which these problems can be fixed. To do this, we define a quality model to implement the proposed framework.

The rest of this paper is structured as follows. Section 2 discusses the existing quality models for usability evaluation. Section 3 provides an overview of the OO-Method and the OlivaNova Model Execution technology. Section 4 explains our strategy for early usability evaluation as well as the underlying quality model.

Section 5 discusses the benefits of our strategy. Finally, section 6 presents the conclusions and future work.

REVIEW OF EXISTING QUALITY MODELS

A quality model is a very useful tool for quality requirement engineering as well as for early evaluation and control of quality. It is defined as the *set of characteristics and the relationships between them which provide the basis for specifying quality requirements and evaluating quality*. Several quality models for usability evaluation have been proposed in the last few years. These models have been defined based on existing standards such as ISO/IEC 9241-11 [13] and ISO/IEC 9126 [14] or defined from scratch. The ISO/IEC 9126-1 [14] provides a software product quality model that is intended to be used as a general-purpose standard quality model. The ISO 9241-11 explains how usability can be specified in terms of user performance and satisfaction. Although these models are useful in providing principles and recommendations, they are generic and need to be extended and or decomposed for their use on different kind of systems.

Quality models defined from scratch include the proposals of Dromey [6], Dix [5], and Nielsen [22]. Dromey [6] uses a constructive strategy to characterize *behaviours* and *uses*. *Behaviour* is defined as something that the software itself exhibits when it executes under the influence of a set of inputs (e.g., usability). A *use* is something that different users do *with* or *to* software. In order to define the model, Dromey follows a bottom-up construction approach that enumerates specific properties and classifies them as pertaining to certain software characteristics, and further enumerates software characteristics that characterize each behaviour and use. A top-down construction approach is also used since Dromey employs derivation and/or decomposition to characterize or to define abstract properties (*behaviours* and *uses*) in terms of subordinated behaviours, subordinated uses, and software characteristics.

The model proposed by Dix [5] is defined according to three main factors: *learnability*, *flexibility*, and *robustness*. These factors are subdivided into sub-characteristics that influence the concept that they belong to. The model proposed by Nielsen [22] is more detailed than the Dix model. It focuses on social acceptability and practical acceptability. Usability is considered as a sub-characteristic of usefulness, which is, in turn, a sub-characteristic of practical acceptability. The usability dimension of the model incorporates the following attributes: *easy to learn*, *efficient to use*, *easy to remember*, *fewer errors*, and *subjectively pleasing*. The first four attributes represent the quality characteristics of a software product, whereas the last attribute (subjectively pleasing) represents end-users' subjective evaluations of a software system. A similar model that only differs from the Nielsen model in the terminology was proposed by Shneiderman [30]. Shneiderman calls his definition of usability "five measurable human factors that are central to the evaluation of human factor goals". These factors are: *speed of performance*, *time to learn*, *retention over time*, *rate of errors by users*, and *subjective satisfaction*.

Alternatively, there is another category of quality models that have been defined based on existing quality standards. The proposals of Van Welie [31], Abran [1], and Fitzpatrick and Higgins [7] are in this category. Van Welie [31] proposes a model that is structured in three layers. In the highest layer there are three aspects that define usability in the ISO 9241-11: *efficiency*, *effectiveness*, and *satisfaction*. The second layer contains a set of *usage indicators* that are indicators of the usability that can be observed in practice when the users interact with the system (*learnability*, *errors/safety*, *satisfaction*, *performance speed*, and *memorability*). Finally, the lowest layer contains the *means*, which cannot be observed in user tests, but can be used in heuristics to improve one or more usage indicators.

Some other proposals try to combine the ISO 9241-11 and the ISO/IEC 9126-1. For example, Abran [1] considers the ISO 9241-11 as the basis for his model and integrates other usability characteristics (learnability and security) from the ISO/IEC 9126 and other sources such as ISO 13407 [16]. The proposed enhanced model follows the ISO/IEC 9126 three-layer structure (characteristics, sub-characteristics, and measures). Thus, the measures proposed by the authors have been analyzed and a recommended set chosen. For instance, effectiveness measures are the percentage of tasks accomplished, the ratio of failure of handling, and the percentage of tasks achieved per unit of time.

Other proposals are based on the quality factors proposed by McCall *et al* [17]. For example, Fitzpatrick and Higgins [7] identified problems that are related to the actual definitions of software usability and models such as the lack of consistency of the software attributes and the definitions that do not support a universal set of measures. To solve these problems, Fitzpatrick and Higgins proposed a model with basic quality factors that are taken from the McCall model [17]. It has been refined and extended through a methodological analysis of three strands that influence these factors: the software quality strand, the statutory obligations strand and the human-computer interaction strand. The software quality strand is related to quality models and factors as well as standards such as ISO 9000-3 [10], and ISO/FDIS 9000-3 [11]. The statutory obligations strand is concerned with regulations that relate to health and safety issues, particularly those related to the minimum safety and health requirements for work with display screen equipment. Finally, the human-computer interaction strand is concerned with 'good' principles and guidelines for the design of dialogues and with the analysis of equipment and environments for these dialogues.

Discussion

Table 1 shows the different quality models discussed in this section. Each proposal is classified according to the following criteria:

- Performance-based attributes: objective attributes related to the performance of the user in his/her

interaction with the system (e.g. operability and learnability).

- Perception-based attributes: subjective attributes related to the perception of the system by the user (e.g. ease-of-use and satisfaction).
- System facilities attributes: attributes related to the system facilities such as user's help, documentation and installation procedures (e.g. Instalability). The system must to provide these attributes independently of the user interaction.
- Measures: indication of whether or not the quality model provides measures for the proposed attributes.

After reviewing the existing quality models approaches, we concluded that no single quality model copes with all the required usability attributes. As shown in Table 1, all the quality model proposals include performance-related criteria, as these are the most common type of attribute. Dromey [6] is the only proposal that does not consider any perception-related attribute, although there exist well-know questionnaires for that purpose. Almost no proposal explicitly defines attributes related to system facilities. Only the Fitzpatrick y Higgins [7] proposal includes an attribute that leads to system instalability. However, this criterion can also be implicitly addressed by the ISO 9241-11 standard by means of user tasks definition.

Overall, what is missing is a quality model for usability evaluation that includes the three types of attributes as well as metrics that can measure them. We argue that a "good" quality model should address the following concerns:

- **Usability attribute type:** usability is an abstract, non-atomic concept that combines the performance-based and perception-based quality attributes of the user-interaction with the system as well as other attributes related to the system facilities (user help, documentation, installation procedures, etc.). Therefore, the proposed quality model should consider these three types of attributes.
- **Usability vs. application domain:** because some attributes are more critical for certain application domains, different weights can be assigned to the sub-characteristics or attributes of the quality model. For instance, a museum application would not require much training for its use; therefore, the learnability is not a critical characteristic for this application type.
- **Inter-relationship among attributes:** the usability of a software system is not a direct sum of the values of each one of the model attributes. It is necessary not only to reach a certain level for each attribute but also to consider how the different attributes are positively or negatively affected by each other (to identify possible conflicts).
- **Usability vs. user types:** usability also depends on the user type for which the system has been developed. For instance, learnability is more important for novices than for expert users.

Proposal	Performance-based attributes	Perception-based attributes	System facility attributes	Measures
ISO 9241-11 [13]	Efficiency, Effectiveness	Satisfaction	-	No
ISO/IEC 9126-1 [14]	Learnability, Understandability, Operability	Attractiveness, Compliance	-	No
Fitzpatrick y Higgins [7]	External Quality Factors: Functionality, Learnability, Reliability, Safety, Security, Correctness, Efficiency, Adaptability, Interoperability, Suitability	Ease-of-Use	Instalability	No
	Internal Quality Factors: Maintainability, Flexibility, Reusability, Portability, Testability	-	-	No
Abran [1]	Effectiveness, Efficiency, Security, Learnability	Satisfaction	-	Yes
Dromey [6]	Learnability, Transparency, Operability, Responsiveness, Customizability, Command-context sensitivity, Operational directness, Hot-keyed, Consistency	-	-	No
Van Welie [31]	Learnability, Errors/Safety, Performance speed, Memorability	Satisfaction	-	Yes
Dix [5]	Predictability, Synthesizability, Generalizability, Consistency, Dialog initiative, Multi-Threading, Task migratability, Substitutivity, Customizability, Observability, Recoverability, Responsiveness, Task conformance	Familiarity	-	Yes
Shneiderman [30]	Speed of performance, Time to learn, Retention over time, Rate of errors by users.	Subjective satisfaction	-	Yes
Nielsen [22]	Easy to learn, Efficient to use, Easy to remember, Few errors	Subjective pleasing	-	No

Table 1. Review of Quality Models for Usability Evaluation

USER INTERFACE GENERATION WITH OLIVANOVA MODEL EXECUTION

OlivaNova Model Execution™ is the industrial implementation of the automatic software production method called OO-Method [27]. OO-Method is based on the separation of the problem space and the solution space. This positions OO-Method as a methodology to implement tools that follow the recommendations of the

MDA paradigm [18], since it separates the conceptual specification of the applications from its possible software implementations.

The formalism that lies behind OO-Method is OASIS [28], a formal language for the specification of information systems. OO-Method includes the following models:

- *Conceptual Model*: divided into four complementary views: the *Object Model*, which describes the static properties of the system in terms of classes and their relationships; the *Dynamic Model*, which describes the aspects related to the control, valid lives, and the interaction between objects; the *Functional Model*, which describes the semantics associated to the state changes of the objects caused by the execution of events; and, finally, the *Presentation Model*, which allows the abstract specification of the user interface requirements.
- *Execution Model:* establishes the transformation rules of a Conceptual Model to its corresponding software representation in a specific technological platform.

In order to better understand how the usability of an application can be evaluated during the conceptual model phase, we will focus on the view of the Presentation Model that directly affects the user interface.

The Presentation Model

This conceptual model allows the specification of abstract user interfaces by means of a pattern language called Just-UI [19] [20]. As Figure 2 shows, the model is structured in three main levels:

Level 1. Hierarchy of Actions Tree (HAT):

This pattern organizes the functionality that will be presented to the different users who access the system. It is a tree where each intermediate node acts as a container with a label, and each leaf node contains a label and a link to an *interaction unit* (IU).

Level 2. Interaction Units (IUs):

It represents abstract interaction units that the user will interact with to carry out his/her tasks. Four types of UIs can be distinguished: Service IU, Instance IU, Population IU, and Master/Detail IU. A Service IU models a dialogue whose objective is to help the user execute a service. An *Instance IU* models the data presentation of an instance and supports interaction with it. It is defined in a class and provides a *display set* (information to be shown), *actions* (services execution) and *navigations* (reachability between instances).

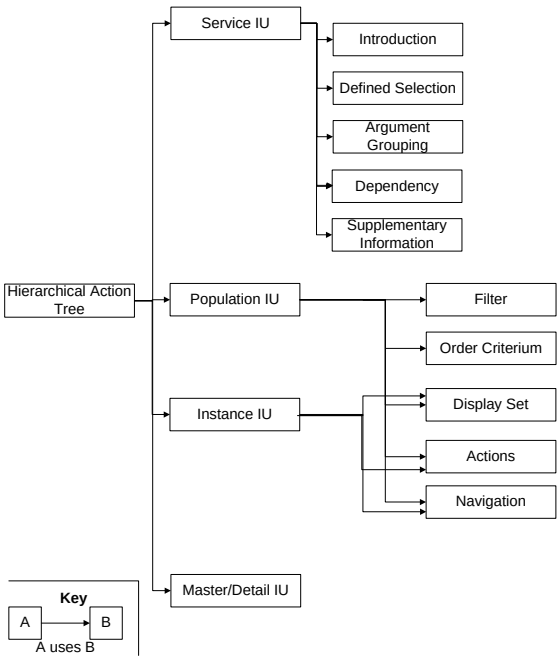


Figure 2. Presentation patterns at the problem space level

A *Population IU* models an interaction unit by showing sets of instances of a class. This pattern deals with object collections. Also, filter and ordering mechanisms can be used to facilitate the selection and consultation of objects. Finally, a *Master/Detail IU* represents more complex interaction units that deal with master/slave presentations. For this reason, the components are divided into two logical types: master components and detail components. These components are related by means of an aggregation relationship in a given direction: from master to detail. Therefore, when a master component changes, the detail component also changes.

Level 3. Elementary patterns (EPs):

These patterns constitute the primitive building blocks of the UI and allow the restriction of the behavior of the different interaction units. They are the following:

- *Introduction:* captures the main features of data types that the user will introduce into the system. It includes the definition of edition masks, a value by default, a rank of values, help messages and the validation messages in case of error.
- *Defined Selection:* enumerates the possible values for a class attribute, a data-valued argument, or a data-valued filter variable. It restricts the arguments used to specify values.
- *Argument Grouping:* organizes the arguments of a service into different groups to facilitate the introduction of data. It is not necessary to display all groups at once.
- *Dependency:* establishes dependency relationships between the arguments of a service. It is an event-condition-action rule. The event indicates the action happening to an argument; the *condition* is a boolean well-formed formula; and the *action* is a well-formed

formula specifying what happens to other arguments of the service when an event happens and a condition holds.

- *Supplementary Information:* captures the additional information shown when a user selects an object. It helps the user to confirm the object selection.
- *Filter:* defines selection criteria and allows obtaining the instances that fulfill the constraint expressed by its formula.
- *Order Criteria:* defined when instances have to be ordered by the value of an attribute or a set of attributes.
- *Display Set:* composed by all attributes to be shown in an abstract interaction unit.
- *Actions:* allows the user to execute actions.
- *Navigation:* allows access to instances of a related class. As navigation leads the user to another I.U., before defining navigation, at least one IU (the one that the navigation reaches) has to be defined.

The following section introduces our model-driven architecture approach for usability evaluation of user interfaces based on the Presentation Model.

OVERVIEW OF THE STRATEGY

Figure 3 presents the framework proposed for early usability evaluation [26]. According to the MDA directives fulfilled by OlivaNova™ [24], the usability of a software system can be evaluated and improved at the conceptual model level taking into account two perspectives: the *problem space* and the *solution space*.

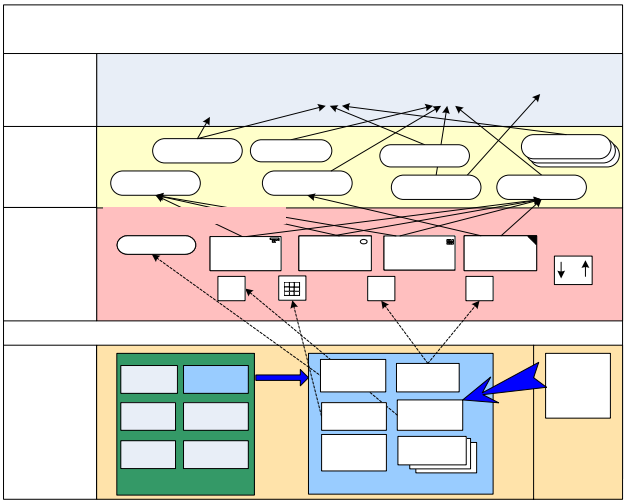


Figure 3. Usability framework (adapted from [8])

Usability in the Problem Space

Usability at this level is defined through mappings that are implemented in three layers: usability definitions, usability criteria, and presentation patterns. The first layer is concerned with the definition of usability.

In the ISO/IEC 9126-1 [14], usability is defined as "the capability of the software product to be understood, learned, used, and attractive to the user, when used under specified conditions". On the other hand, the ISO 9241-11 [13] 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". This last definition is in agreement with the definition of the *quality in use* term described in Part 4 [15] of the ISO/IEC 9126.

Although both usability views are important and necessary in the development of a software process, we believe that the ISO 9126 view is more appropriate for our framework since our goal is to evaluate usability at the conceptual model phase *without* considering the use of the product by potential users in specific use contexts. Note that the ISO 9241-11 view on usability is focused on the product effect (efficiency, effectiveness, and user satisfaction) at the time of use of the product, whereas the ISO 9126 view is focused on product attributes derived by the user interface and interaction.

According to ISO/IEC 9126, usability is a quality characteristic that includes the following sub-characteristics: understandability, learnability, operability, attractiveness, and compliance. These concepts are abstracts and cannot be directly measured. In order to decompose these sub-characteristics into measurable attributes, we have analyzed several usability patterns and ergonomic criteria proposed in the literature [3] [8] [9] [30]. From the numerous lists of usability attributes and ergonomics criteria, we have defined a *quality model* that supports the usability framework. This model establishes the relationship among usability attributes and the properties of the presentation patterns. The purpose is to determine which presentation patterns contribute to the satisfaction of certain usability attributes. This quality model is introduced in the next section.

Usability in the Solution Space

The different code generation strategies of the OlivaNova™ Model Execution technology are located in the solution space. These code generators allow us to obtain a complete three-tiered (business logic, interface and database) software application in the following platforms: Visual BASIC, Java/Swing, ColdFusion, JSP, ASP, and PocketPC.

During the automatic generation of user interfaces, specific aspects of the domain are taken into account. Depending on the code generator that is applied, the abstract presentation patterns (AOI - Abstract Object Interface) are instantiated in the corresponding software components in the target platform architecture (COI - Concrete Object Interface). This strategy is called reification [32]. COIs are dependent on a given implementation, whereas AOIs are control abstractions that are independent of implementation and design considerations. The duality between COIs and AOIs has allowed us to abstract usability attributes and other relevant properties from the user interface elements.

We also verified that some usability attributes are fulfilled during the reification process. The following *user guidance* attributes related to visual feedback are directly ensured: (1) visual feedback in menus or dialog

boxes where choices can be selected, (2) visual feedback in menus or dialog boxes where the cursor is pointing, (3) visual feedback where options are already selected for multiple options, (4) visual feedback when objects are selected. In addition, other attributes related to the *legibility* of the user interface are also ensured: (1) prompts and error messages always appear in the same UI place(s), (2) error messages displayed in pop-up windows indicate the field(s) where the error has occurred.

QUALITY MODEL

We define in this section a quality model that addresses the concerns listed in the review of existing quality models section. The development of the proposed quality model consists of the following activities: define quality goals, specify quality characteristics, specify relationships, and operationalize the model. These activities are described below.

Define Quality Goals

In this activity, the characteristics that are to be covered by the evaluation and other issues of quality modeling such as the object (artifact) and the stakeholder (viewpoint) should be clearly defined. According to the Goal-Question-Metric (GQM) paradigm [2], the goal of our quality model is:

	Description	Definition
Object	Which artifact should be analyzed?	Analyze the *Presentation Model*
Purpose	Why should the object be analyzed?	For the purpose of *evaluation*
Quality focus	Which object characteristics should be analyzed?	With respect to its *usability*
Viewpoint	Who will use the data collected?	From the viewpoint of the researcher
Context	In which environment does the analysis take place?	In the context of CARE Technologies, S.A.

Table 2. Quality model goal definition

Specify Quality Characteristics

This activity is concerned with the decomposition of the quality focus characteristics (i.e., usability) into a set of quality sub-characteristics and measurable attributes. An attribute is measurable when it is possible to define one or more corresponding metrics.

In order to do this, the ISO/IEC 9126-1 quality model is used as the basis for building the proposed usability model. In this model, the following sub-characteristics are identified:

- **Learnability:** the capability of the software product to enable the user to learn its application.
- **Understandability:** the capability of the software product to enable the user to understand whether the software is suitable, and how it can be used for particular tasks and conditions of use.
- **Operability:** the capability of the software product to enable the user to operate and control it.
- **Attractiveness:** the capability of the software product to be attractive to the user.
- **Compliance:** the capability of the software product to adhere to standards, conventions, style guides, or regulations related to usability.

All these high-level characteristics can be decomposed into more detailed sub-characteristics and attributes based on the user-interaction or other attributes concerned with help features, user documentation, and installation instructions. Actually, such a decomposition must be done in order to support usability inspection methods and to identify and fix usability problems.

We argue that the first three sub-characteristics (learnability, understandability, and operability) are related to the performance of the user using the system and they can be quantified using objective measures. *Learnability* refers to the attributes of a software product that facilitate learning. It also corresponds to the attributes for "suitability for learning" as defined in ISO 9241-10 [12]. In our model, this sub-characteristic consists of *help facilities* (pre-defined keyboards, wizards, on-line help, and documentation) as well as *predictability* which refers to the ease with which a user can determine the result of his/her future actions, and *informative feedback* in response to user actions.

Understandability refers to the attributes that facilitate understanding. It can be decomposed into user interface *legibility*, which is mainly affected by the *information density and information grouping cohesiveness* as well as other attributes such as brevity, UI *self-descriptiveness*, *messages quality* and *navigability*. For instance, *brevity* is mainly related to patterns that reduce the cognitive effort of the user, and navigability is related to guiding the user by providing mechanisms such as path and current position. In our model, *navigability* is measured in terms of the absence of cyclic navigations among abstract UIs, which avoids the phenomenon of "lost in the hyperspace".

Operability refers to the attributes that facilitate user control and operation. It also corresponds to controllability, error tolerance, and conformity with user expectations as defined in ISO 9241-10. In our model, this sub-characteristic is concerned with the capacity of the system to provide: data validation (*data validity*), the user's degree of control of the service execution (*controlability*), the degree of adaptability of the user interface or services (*adaptability*), the operational consistency in the execution of services and controls (*consistency*), the prevention and management of errors (*error prevention*), the significance of a label and the items or actions they refer to (*labeling significance*), the capacity to *monitor the state of the system*, and user *accessibility*. Accessibility is more applicable to web domains and can be measured by verifying the proportion of presentation patterns in accessible format (*completeness*) as well as the proportion of presentation

patterns that are adapted in accordance with user disabilities (*adaptability*). It represents more than one way to represent, display, and enter data.

Finally, the last two sub-characteristics are related to the perception of the end-user (*attractiveness*) or evaluator (*compliance*) using the system and can be measured using subjective measures. However, we argue that some aspects of attractiveness related to aesthetic design can be quantified by measuring the UI uniformity in terms of font color, font style, font size and elements position. Some studies suggest that designs should not use more than 2-3 fully saturated intense colors.

Also, the *compliance* sub-characteristic can be measured by assessing the agreement of the proposed quality model with respect to the following standards: ISO/IEC 9126 (parts 1 and 3), ISO 9241 (part 10), and the Microsoft style guide. The first two assessments are centered at the problem space level, whereas the last one is centered at the solution space level as it analyzes the fulfillment of the software components of the different generation strategies with respect to the prescriptions of the Microsoft style guide.

The complete usability decomposition is described below:

1. **Learnability**
   - 1.1. Help Facilities
   - 1.2. Predictability
     - 1.2.1. Icon Significance
     - 1.2.2. Icon/Link Title
     - 1.2.3. Action Determination
   - 1.3. Informative Feedback
2. **Understandability**
   - 2.1. Legibility
     - 2.1.1. Information Grouping Cohesiveness
     - 2.1.2. Information Density
   - *2.2.* Workload
     - 2.2.1. Brevity
     - 2.2.2. Self-descriptiveness
   - 2.3. User Guidance
     - 2.3.1. Message Quality
     - 2.3.2. Navigability
3. **Operability**
   - 3.1. Data Validity
   - 3.2. Controlability
   - *3.3.* Adaptability
   - 3.4. Consistency
     - 3.4.1. Steady Behavior of Controls
     - 3.4.2. Permanence of Controls
     - 3.4.3. Stability of Controls
     - 3.4.4. Order Consistency
     - 3.4.5. Label Consistency
   - 3.5. Error Prevention
   - 3.6. Labeling Significance
   - 3.7. State System Monitoring
   - 3.8. Accessibility
     - 3.8.1. Completeness
     - 3.8.2. Adaptability
4. **Attractiveness** (Solution Space)
   - 4.1. Font Color Uniformity
   - 4.2. Font Style Uniformity
   - 4.3. Font Size Uniformity
   - 4.4. UI Position Uniformity
5. **Compliance**
   - 5.1 Degree of fulfillment with the ISO/IEC 9126-1 and ISO/IEC 9126-3
   - 5.2 Degree of fulfillment with the ISO 9241-10
   - 5.3 Degree of fulfillment with the Microsoft style guide.

Specify Relationships

After defining usability characteristics, sub-characteristics, and attributes, the relationships among them can be defined. The relative importance of the characteristics and attributes can be determined as well as the way in which they positively or negatively affect each other. Asking experts to weigh the relative importance of usability characteristics is one way to identify the set of the most important attributes.

Another important task in this activity is the association of one or more properties of the presentation patterns to each identified attribute. This is due to the fact the attributes in our model are inherent to user interface modeling and make a product usable. Table 3 shows several examples of the relationships established between a usability *attribute* and a *presentation pattern property*. For instance, the *Data Validity* attribute is related to the following properties of the *Introduction* pattern: edition mask, value range, and maximum size. Similarly, the *Labeling Significance* attribute is related to the existence of the meaningful alias defined as part of the Introduction, Defined Selection, Ordering Criteria, Filter, and Action patterns.

Operationalize the Model

This step consists of quality model quantification. We associate one or more *metrics* to each *attribute* of the quality model [25]. The model can combine quantitative and qualitative measures. Table 3 shows examples of potential metrics for some usability attributes of the model.For instance, a potential metric for the *Message Quality* attribute is the *Proportion of meaningful error messages*. An error message is meaningful if it informs the user of the nature of the error, the probable cause and the way to fix it. For instance, the following kind of message must be avoided: Item not found.

Sub-characteristic	Attribute	Presentation patterns properties affected	Rationale	Potential metrics
Legibility	Information Grouping Cohesiveness	HAT/actions grouping, Service IU/ argument grouping.	Actions can be grouped to build user menus using criteria such as hierarchic organization and sorting (alphabetical, by class the actions belong to, etc.). The arguments can also be grouped	Proportion of actions grouped by class Proportion of actions grouped by related classes

User Guidance	Message Quality	Introduction (help messages), Defined Selection (help messages)	The quality of error messages promotes users' understandability. The help messages of the patterns must indicate to the users the reasons for their errors, their nature, and the way to solve their errors.	Proportion of meaningful error messages Proportion of meaningful warning messages
Operability	Data Validity	Introduction (edition mask , value range, and maximum size)	This pattern addresses the problem of detecting and preventing data entry errors or actions. For instance, the *value range* represents the maximum and minimum values that can be introduced, and *maximum size* is the number of characters allowed for an item.	Number of defined properties errors per IU Proportion of patterns in an IU that applies data validation Proportion of patterns in the model that applies data validation
	Labeling Significance	Introduction (alias), Supplementary information, Defined Selection (alias), Ordering Criteria (alias), Filter (alias) Navigation, and Actions (alias)	A label is significant to the users when there is a strong semantic relationship between such labels and the items or actions they refer to.	Proportion of elements with meaningful names
	Order Consistency	Population IU, Navigation, Action, Service IU	The order of the different presentation patterns (i.e., services, actions, navigations) must always be the same across the IUs.	Proportion of actions in the same order across related IUs Proportion of navigations in the same order across related IUs

Table 3. Sub-characteristics, attributes, presentation patterns and potential metrics

On the other hand, the *Order Consistency* attribute is measured by verifying the following: (1) the order of the arguments in the services of the same class or related classes (e.g., the create and modify services must to have its arguments in the same order); (2) the order of the attributes in the display set of the same or related classes; (3) the order of the navigation patterns; (4) the order of the services in the action pattern of a class or related classes (e.g., the services create, modify and delete must to appear in the same order across UIs).

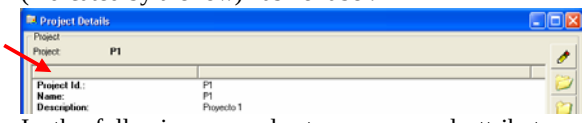
EXAMPLES

The development and use of the quality model has allowed us to identify usability problems at the problem and solution space levels as well as to identify missing usability attributes in the Presentation Model. These problems are briefly discussed below. For reasons of simplicity, we explain the usability problems using the following scheme: *problem* is the usability problem description; *context* is the pattern(s) of the Presentation Model where the problem was observed; *related usability criteria* are the criteria affected by the problem; *usability sub-characteristic* are the ISO/IEC 9126 usability sub-characteristics that are affected; *example* is an example that illustrates the problem; and a possible *solution* for the problem.

Usability Problems in the Problem Space

Some of the usability problems that we observed when defining the proposed quality model are described below. In order to facilitate comprehension, we provide excerpts of user interfaces which were automatically generated from the analyzed conceptual models.

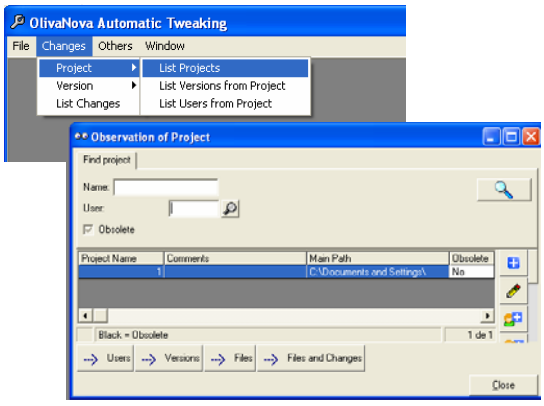
- **Problem:** Lack of descriptive labels
  Context: Instance IU and Supplementary Information.
  Related usability criteria: Guidance
  Usability sub-characteristic: Labeling Significance
  Example: Note that the display set of the Instance IU (indicated by the row) has no label.



In the following example, two composed attributes defined as part of a supplementary information pattern lack a label:

Solution: Apply the proposed metric described in Table 3. After that, identify the patterns in the model without alias and define them.

- **Problem:** Lack of consistency between a label in the HAT and the label of the interact unit associated to it.
  Context: HAT, Service IU, Instance IU, Population IU, and Master-Detail IU.
  Related usability criteria: Consistency
  Usability sub-characteristic: Label Consistency
  Example: The *List Projects* label in the HAT has a different label in the IU title (*Observation of*



Solution: Use the same label for a leaf node in the HAT and the title of the interaction unit associated to it.

Missing Usability Attributes

Some attributes of the quality model cannot be evaluated because there is no pattern in the Presentation Model that represents them. This is the example of the *Adaptability* and *Accessibility* attributes of the operability sub-characteristic and the *Action Determination* attribute of the learnability sub-characteristic. They are described following.

- **Problem:** There is no way to adapt the information to be presented according to the users' needs and preferences.
  Context: HAT, Population IU, Introduction, Action.
  Related usability criteria: Adaptability, Flexibility, User Experience
  Usability sub-characteristic: Operability
  Solution: Extend the Presentation Model by including adaptation issues (i.e., HAT adaptation depending on the user experience).

- **Problem:** The user needs to know possible actions and navigations with the purpose of determining his future action. Currently, all actions are available and if the user chooses an action not available in a given IU an error message is displayed.
  Context: Instance IUs, Population IUs, Master-Detail IUs, Actions, Navigations.
  Related usability criteria: User Explicit Control, Predictability
  Usability sub-characteristic: Operability
  Solution: Extend the Presentation Model by allowing the definition of different alternatives to show information (i.e., activate/deactivate actions and navigations depending on the data to the presented.

Usability Problems in the Solution Space

We also observed some problems that are related to the implementations of the different transformation engines. For instance, the help reference is not linked to the generated application and the F1 functionality on controls are not implemented. Therefore, the *Help Facilities* attribute of the learnability sub-characteristic cannot be measured. A potential metric for this attribute could be the number of tasks with online help. Another problem observed at the code generators level was the following:

- **Problem:** The data of a display set (e.g., grid) is not refreshed when a service (i.e., create) is executed.
  Context: Population IU, Master/Detail IU
  Related usability criteria: Guidance, current information visualization.
  Usability sub-characteristic: User Guidance
  Example: The data of a master grid is not refreshed when a service is executed in the detail part of a Master/Detail IU.
  Solution: Refresh data after a service execution checking whether the action was triggered from a Population IU or from a Master/Detail IU.

CONCLUSIONS AND FUTURE WORK

We have presented the definition of a framework for early usability evaluation. Unlike existing proposals, the usability of a system is evaluated and improved at the conceptual model level. This framework has been operationalized by means of a *quality model* which represents the relationships among the ISO/IEC 9126 usability sub-characteristics, the usability patterns and criteria proposed in the literature, and the ONME presentation pattern properties. The result is a set of measurable usability attributes.

This usability framework allows us to identify usability problems at both the problem space and the solution space level. Usability problems identified at the problem space level can be fixed by improving the Presentation Model specification. In this model, we also have found a lack of expressiveness to fulfil some usability attributes (i.e., adaptability and accessibility). Currently, we are applying the quality model to several real-life projects in CARE Technologies to perform quality evaluations early in the development process, to learn effectively across several product releases, and to refine the quality model through subsequent projects. Further research includes the empirical evaluation and improvement of the proposed quality model and the automation of the evaluation of some attributes.

REFERENCES

1. Abran A., Khelifi A., Suryn W., Seffah A. (2003) Usability Meanings and Interpretations in ISO Standards, Software Quality Journal, 11 (4) 325-338.
2. Basili, V., Rombach, H. (1988) The TAME Project: Towards Improvement-Oriented Software Environments. IEEE Transactions on Software Engineering 14 , 758-773.
3. Bastien, J. M. and Scapin, D. L. Ergonomic Criteria for the Evaluation of Human-Computer Interfaces, version 2.1, 1993.
4. Constantine, L., Lockwood, L. A. D. (1999), Software for Use: A Practical Guide to the Models and Methods of Usage-Centered Design, Addison-Wesley, NY.
5. Dix, A., Abowd, G., Beale, R. and Finlay, J. (1998), Human-Computer Interaction, Prentice Hall Europe.
6. Dromey R.G. (1998) Software Product Quality: theory, Model and Practice, Software Quality Institute, Griffith University, Nathan, Brisbane, Australia.
7. Fitzpatrick R., Higgins C. Usable Software and its Attributes: A synthesis of Software Quality, European Community Law and Human-Computer Interaction.
8. Folmer E., Bosch J. (2004) Architecting for usability: A Survey, Journal of Systems and Software, 70(1) 61-78.

9. Hix, Deborah and H. Rex Hartson (1993), Developing User Interfaces: Ensuring Usability Through Product & Process. New York, NY: John Wiley & Sons.
10. ISO 9000-3 (1991) Quality management and quality assurance standards - Part 3: Guidelines for the application of ISO 9001 to the development, supply and maintenance of software, ISO, Switzerland.
11. ISO/FDIS 9000-3 (1997) Quality management and quality assurance standards - part 3: Guidelines for the application of ISO 9001:1994 to the development, supply, installation and maintenance of computer software, ISO, Switzerland.
12. ISO 9241-10 (1996) Ergonomic requirements for office work with visual display terminals (VDTs) -- Part 10: Dialogue principles.
13. ISO 9241-11 (1998) Ergonomic requirements for office work with visual display terminals - Part 11: Guidance on Usability.
14. ISO/IEC 9126-1 (2001), Software engineering - Product quality - Part 1: Quality model.
15. ISO/IEC TR 9126-4 (2004), Software Engineering - Product quality - Part 4: Quality in use metrics
16. ISO/IEC 13407 (1999), User centred design process for interactive systems. International Organization for Standardization, Geneva.
17. McCall J. A., Richards P. K. and Walters G. F. (1977) Factors in software quality, Vols. IIII, Rome Aid Defence Centre, Italy
18. MDA: http://www.omg.org/mda.
19. Molina P., Especificación de interfaz de usuario: De los requisitos a la generación automática, PhD Thesis, Dept. de Sistemas Informáticos y Computación. Universidad Politécnica de Valencia, Marzo de 2003.
20. Molina, P. J. Meliá, S. and Pastor, O. (2002), Just-UI: A User Interface Specification Model. In Ch. Kolski and J. Vanderdonckt (Eds.), Computer-Aided Design of User Interfaces III, Kluwer Academics Publisher, 63–74.
21. Myers, B. A., Rosson, M. B. (1992), Survey on User Interface Programming. In Striking a Balance. Proc. of CHI'92. May 1992, New York: ACM Press, 195-202.
22. Nielsen, J. (1993), Usability Engineering, Academic Press, London.
23. Nielsen, J., 1994. Heuristic Evaluation, in Usability Inspection Methods. Nielsen, Jacob and Mack,R. L., John Wiley and Sons, New York, NY.
24. OlivaNova, http://www.care-t.com/products/index.html
25. Olsina L., Martín M. (2004) Ontology for Software Metrics and Indicators, Journal of Web Engineering, Rinton Press, Vol 2 Nº 4, pp. 262-281, ISSN 1540-9589.
26. Ortiz, J. C., Condori-Fernandez, N., Abrahão S., A (2005) Usability Framework for the Applications Generated with the OlivaNova Model Execution technology, 6th Spanish Congress on Human Computer Interaction (Interacción 2005), Granada, Spain.
27. Pastor, O., Gómez, J., Insfrán, E. Pelechano, V. (2001) The OO-Method Approach for Information Systems Modelling: From Object-Oriented Conceptual Modeling to Automated Programming. *Information Systems*, 26(7) 507–534.
28. Pastor O., Hayes F. y Bear S., OASIS: An object oriented specification language. CAiSE 92, LNCS 593, Springer-Verlag, pag.348-363; 1992, Manchester (UK).
29. Preece, J., Rogers, Y., Sharpe, H., Benyon, D., Holland, S. and Carey, T. (1994) *Human-computer interaction*, Addison-Wesley, Wokingham, UK.
30. Shneiderman, B. (1998), Designing the User Interface: Strategies for Effective Human-Computer Interaction, Addison-Wesley, USA.
31. Van Welie, M., Van der Veer. G. C., Eliëns A. Breaking down Usability.
32. Vanderdonckt J. (1993), Encapsulating knowledge for intelligent automatic interaction objects selection. Proc.of the SIGCHI conf. on Human factors in computing systems.
33. Wixon, D., Wilson, C. (1997), The Usability Engineering Framework for Product Design and Evaluation. I, In Handbook of Human-Computer Interaction. Helander, M. G., Elsevier North-Holland.

The Applicability of Software Quality Standards to Digital Libraries: A Work in Progress

Effie Lai-Chong Law
ETH Zürich
Gloriastrasse 35
law@tik.ee.ethz.ch

ABSTRACT
The applicability of the prevailing software quality standards to insuring the utility and usability of Digital Library (DL) cannot be taken for granted, because of the vast heterogeneity of its end-users, the complexity of information organization, and the out-of-sync standards engendered by lengthy ratification processes. In this paper, we selected three standards (ISO/IEC 9126-1, 9241-11, 19796-1) to assess their adequacy in addressing specific problems of academic DLs. Preliminary results reveal the limitations of the standards, especially the definitional issue of the key notion 'goal' and relational as well as computational issue of usability parameters. Implications for refining the standards are drawn.

Author Keywords
Quality standards, Digital library, Metrics, Bibliographic principles

ACM Classification Keywords
H5.m. Information interfaces and presentation (e.g., HCI): Miscellaneous.

1. INTRODUCTION

A *standard* is a framework of specifications that has been approved by a recognized standards organization (de jure standard), is accepted as a de facto standard by the industry or belongs to open standards. Specifically, in the field of Human-Computer Interaction (HCI) and Software Engineering (SE), a number of standards addressing software design and evaluation are available. Nonetheless, the applicability of these standards to the ever-changing Information Technology (IT) products cannot be taken for granted, considering that these standards normally need to go through lengthy ratification processes and thus may not be able to keep in sync with the rapid IT development. Further, innovative IT leads to an escalation of new opportunities for augmenting, extending and supporting learning and teaching in a diversity of contexts.

In the recent decade e-Learning has drawn a lot of research as well as practical concerns and efforts in the academic community and industry. Digital Library (DL) is seen an integral part of online education. DL is broadly defined as "*information systems (IS) and services that provide electronic documents – text files, digital sound, digital video – available in dynamic and archival repositories*" [6, p.1023]. Alternatively, DLs are also defined in terms of "component repositories", "knowledge networks" [34] or "structured collections of validated information" [3]. One of the key determinants of the quality of DLs is *search effectiveness*, though how to define this parameter precisely is controversial. There exist a variety of DLs with some being developed for private commercial companies and others for academic institutions and public bodies. Yet the potential of DLs has not (fully) realized. One of the limiting factors is their poor usability. A new line of research to address this specific issue has been launched in recent years [e.g. 2, 28]. Given a wide range of application contexts, user experiences, design methods as well as technological infrastructures, pluralistic evaluation methods are required to assess the utility and usability of DLs. Linking DLs across countries and rendering them interoperable as well as seamlessly integrated is a new technical challenge, especially when end-users of such large and complex federated DLs are widely distributed and highly diverse.

This paper aims to explore a research question: *To what extent can existing software quality standards support the design and usability evaluation of Digital Libraries?* For this purpose, we have identified three standards which explicitly address usability in addition to other software quality attributes: ISO/IEC 9241-11 [13], ISO/IEC 9126-1 [14], and ISO/IEC 19796 [17]. The former two have widely been disseminated and adopted, whereas the latter has lately been developed and relatively less well-known. The applicability of these standards to addressing the usability of DLs will be evaluated on the conceptual and empirical levels. Note that this is a work in progress and thus only limited data have been obtained.

2. SOFTWARE QUALITY STANDARDS IN HCI AND SE

Three software quality standards have been selected because of their wide adoption, high popularity or recency. The DLs of interest are known as **EducaNext**, **ELENA-HCD Suite (beta version)**, **EdNA Online, MERLOT,** and **eduSource.** They have been selected based on the following reasons: They comprise

component repositories supporting *federated search*[1]; it is in the public domain and managed by a non-profit organization; the data required for the analysis are (partially) accessible. Subsequently, we review the three standards with reference to the data on these DLs

2.1 ISO/IEC 9241-11 (1998)

The standard 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."* Furthermore, the three usability metrics are defined as follows:

- Effectiveness – the accuracy and completeness with which users achieve specified goals
- Efficiency – the resources expended in relation to the accuracy and completeness with which users achieve goals
- Satisfaction – the comfort and acceptability of use

To operationalize the terms, effectiveness and efficiency are the function of unassisted task completion rates and task completion times, respectively. While these metrics may be valid for e-Commerce websites of which the usage is normally discretionary and thus long downloading time will drive users away, they may be invalid for academic DLs. A common usage scenario can well illustrate the point. Whether DL users are asked to locate known items or some items relevant to topics of interest, it is highly probable that the search result will modify their needs and goals, especially when they locate extra items that were not originally included as target. For instance, a junior postgraduate student intends to work on a project related to "Online Communities", which she keys in the search engine of a DL. More than 1000 records are returned. She scans the first few pages of records and thus becomes aware that Online Communities is an umbrella term subsuming a large number of sub-topics. After considering for a moment, she refines her topic as "Online Communities for the Disabled" and uses these topical words to re-start the search. In this case, it is difficult to define the cutting point for task completion.

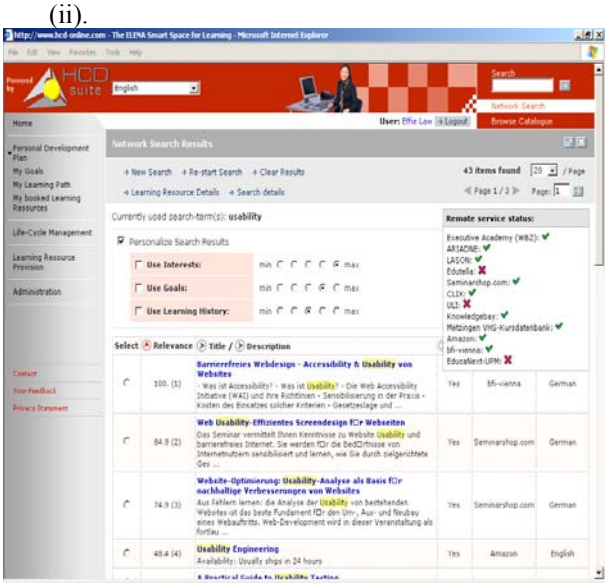
Some attempts to refine these metrics (cf. 'search efficacy' [19]) have been made. However, these metrics are not single-dimensional; combinatorial measurements taking all contributing contextual factors into account are yet to develop. Indeed, identifying such factors is already a challenge let alone translating them into computational terms. For instance, it was shown that users' searching behaviour would vary substantially with the testing environment (e.g., with or without the presence of an experimenter), especially when the searching task was open-ended, i.e., no constraint on specific topical areas, no time limit, etc. [e.g. 30].

2.1.1 Federated Search with ELENA

ELENA Human Capital Development (HCD) Suite (http://www.hcd-online.com/HCDExp/ubp) comprises a *personalized* search component that enables a learner to find in a network of educational repositories those learning resources that align the learner's goals and interests.

***Design*:** In the context of DLs user goal can be very ill-defined and incoherent, especially when users explore a completely new topical area. Consequently, usability becomes a fuzzy notion if the reference point – user goal – is floating. With the advent of Semantic Web technology, it is possible to include more semantic information in digital resources [32], enabling DLs to create new forms of personalization. The ELENA project has developed a prototype of search engine that aligns ranking of learning resources found with user goals and interests (Figure 1). A series of user trials with a number of search tasks have been conducted [21]. Users had to go through the following procedures:

(i) Describe 3 goals and 3 interests in a free-text format;
(ii) Perform six compulsory and some optional search tasks by specifying a search term (free text format) and a cluster of query attributes (e.g. language and price of learning resource), *with* and *without* personalization effects
(iii) Assess the consistency of the system-generated ranking with the user-defined ranking of the learning resources found as a result of the search performed in (ii).



[1] Federated search is defined as "*support for finding items that are scattered among a distributed collection of information sources or services, typically involving sending queries to a number of servers and then merging the results to present in an integrated, consistent, coordinated format*" [1]

Figure 1: ELENA user interface for personalized search

Results: Preliminary results showed that on average personalization of the ranking had almost no effect. Interestingly, personalized ranking improves search results if a search term is general, but has negligible effect if a query is too specific. In case a learner has multiple goals on very diverse topics in her profile, ranking results become slightly worse. These observations may be attributed to the ineffectiveness of the chosen algorithms and to the inconsistency of user cognition. The former addresses the technical aspect of the system design, is somewhat corrigible, and can further be evaluated by benchmarking the system with other personalized search engines on the same set of educational nodes, whereas the latter addresses the psychological aspect of the system usage, is somewhat uncontrollable, and can further be evaluated with carefully well-designed experimental studies.

Discussion: The findings of the ELENA empirical studies seem to imply that user goals – the key notion in ISO/IEC 9241-11 – may not be a valid reference point for evaluating usability, given low goal-based personalization effect and apparent inconsistency of user goals. From the theoretical and philosophical point of view, psychological constructs such as goal, plan and intentionality are somewhat elusive, because they can be transient, contingent and emergent. Indeed, the nature of goal is disputable between the three approaches to the study of context, namely Situated Action (SA), Activity Theory (AT) and Distributed Cognition (DC) [26]. These three approaches have widely been discussed in the field of HCI.

In accord with SA, goals or plans play a minimal role in real actions in context and they are "*retrospective reconstructions*" of the actions done [33]. SA emphasizes responsiveness to the environment and the improvisatory nature of human activity [22]. In contrast, both AT and DC recognize the significant role of goals in regulating the nature of activity. In addition, AT stresses the transformative relationship between people and artefacts with which they interact. Hence, goals as an integral part of context change dynamically through the enactment of an activity. Similarly, DC is concerned with structure – representations inside and outside the head – and the transformations these structures undergo, and with understanding the coordination among individuals and artefacts. However, DC sees goals as inherent property of the system, which may eventually be located in the minds of people who are part of the system. Nonetheless, internal representations (i.e. embodied goals) co-evolve with external representations displaying in artefacts (e.g. search results displayed on the screen). While SA's position of dismissing the role of goals seems rather radical and impractical, AT's and DC's suggestion about the malleability, mutability and 'contextualizability' of goals arouse some compelling concerns:

- How should the term "specified goals" in ISO 9241-11 definition of usability be interpreted[2]? Are they givens like other components of the product under evaluation (user, task, equipment and environment)? The picture is complicated by the fact that different stakeholders may have different goals with respect to the system use. The question then becomes: How compatible are system-imposed, designer-defined, evaluator-defined, and user-defined goals?
- How should the two usability metrics "effectiveness" and "efficiency" be measured if the reference point – goal – is always in flow?
- Is goal-based personalization doomed to fail?

In summary, the two defining criteria of effectiveness - accuracy and completeness – are difficult to estimate in the context of searching with DLs; it is difficult to set the end-point when the search tends to be recursive and it is also ambiguous to estimate the correctness of search results when user goals tend to be mutable.

2.1.2 Usability Tests on EducaNext

EducaNext (http://www.educanext.org/) is a multilingual academic portal supporting the sharing of learning resources for Higher Education. It is open to any members of the academic and research community.

Design: Two sets of usability tests, designated as UT1 and UT2, have been performed on version (v) 0.9 and 1.0 of EducaNext, respectively. Note that v. 0.9 and v.1.0 were similar in terms of functionality and navigational structure, but different in presentational design. For UT1, seven representative users from a university in Switzerland and three users from a research institute in Slovenia were involved. Three and eight representative users from the two same universities took part in UT2. All users were required to fill in a pre-test questionnaire to gather their background data. In both UTs, each user was required to perform ten tasks with nine of them being common. These tasks cover the core functions of EducaNext. Users were instructed to think aloud when working on the tasks. All users' verbalizations and onscreen activities were captured by specific software applications. After each task users were required to complete a three-question "After Scenario Questionnaire" (ASQ) [23]. Further, having finished all the ten tasks they were required to complete a post-test questionnaire "Computer System Usability Questionnaire" (CSUQ) [23]. Objective measures including time-on-task (i.e. efficiency) as well as number of usability problems (UPs)

[2] Timo Jokela (this volume) addresses the issue of "goal" in the ISO 9124-11 definition of usability, but from a different perspective.

identified (i.e. effectiveness), and subjective measures including users' running commentary as well as ratings of ASQ and CSUQ (i.e. satisfaction) were obtained.

Results: We analyzed the relationship between objective measures and subjective measures, i.e., how the three usability measures correlate with each other.

The first two questions in ASQ measured the user's perceived ease and perceived efficiency of completing individual task, using a 7-point Likert scale with left and right anchors of '*Strongly Disagree*' and '*Strongly Agree'*, respectively:

Q1: Overall, I am satisfied with the ease of completing the tasks in this scenario
Q2: Overall, I am satisfied with the amount of time it took to complete the tasks in this scenario

We look at the data of the most complex task – Task 2 (Provide a New Learning Resource, Figure 2). The mean perceived ease and perceived efficiency of completing Task 2 over all the 22 participants were 4.37 and 4.7, respectively.

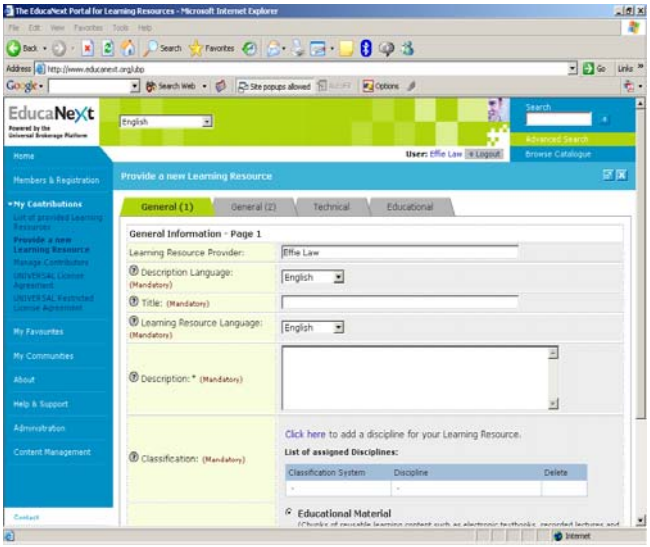


Figure 2: User interface for provision of a new learning resource (Task 2) of UT2 with EducaNext v.1.0

These subjective measures were not consistent with the objective ones. Indeed, there were no significant correlations between the perceived ease of task completion and the number of UPs identified in Task 2. No significant correlations between the perceived efficiency of task completion and the task-completion-time of Task 2 could be found either. These results corroborate those of the previous studies that subjective ratings and objective measures of performance do not necessarily correlate [11, 20, 27, 36], and interestingly the magnitude of such a correlation varies with the users' level of computer experience. To verify this observation, the 22 participants of the present study were categorized into experienced and inexperienced computer users, based on their self-reported level of experience in operating database systems, using a 5-Likert scale with 1 being 'very poor' and 5 being 'very rich'. However, no significant correlation between any of the aforementioned subjective and objective measures could be obtained. Other factors such as level of competence in information and communication technologies (ICT) and experience in e-Learning (i.e. the domain of the system usability tested) did not have any effect on the relationship between the two types of measures either. This particular issue can further be addressed under the framework of Technology Acceptance Model [25].

Discussion: According to ISO/IEC 9241-11 (1998) Section 5.4.1 Choice of Measures "*If it is not possible to obtain objective measures of effectiveness and efficiency, subjective measures based on the user's perception can provide an indication of effectiveness and efficiency*". This statement implies that objective usability measures should significantly correlate with subjective ones. However, our empirical findings tend to refute this implication. Nonetheless, it is speculated that the correlation might be stronger after the learning phase (personal communication, Timo Jokela, 17/08/2005)

To sum up, there are two major issues with ISO/IEC 9241-11 (1998): (i) the inherent fuzziness of the key notion 'specified goal'; (ii) the lack of correlation between objective and subjective usability measures.

2.2 ISO/IEC 9126-1 (2001)

According to ISO/IEC 9126, usability is defined as: *"a set of attributes of software which bear on the effort needed for use and on the individual assessment of such use by a stated or implied set of users."* Interestingly, this definition does not explicitly address the key notion *goals*, as it does in ISO/IEC 9241-11. In addition, 'the effort needed for use' and 'the individual assessment' somewhat correspond to objective (i.e. efficiency) and subjective measures (i.e. satisfaction) specified in ISO 9241-11. Furthermore, usability as defined in ISO 9241-11 depends on software qualities which are distinct from usability as defined in ISO 9126-1.

Specifically, ISO 9126-1 provides a hierarchical quality model comprising six broad categories of Internal and External factors - `usability`, `functionality`, `reliability`, `efficiency`, `maintainability` and `portability`, which are divided into sub-characteristics. It also defines Quality-in-Use using four factors – `effectiveness`, `productivity`, `safety` and `satisfaction`. This quality model is essentially built upon McCall [24] FCM (Factor, Criteria and Metric) 1977 model (see also [7]). Subsequently, we delineate the

quality model of EducaNext and then analyze to what extent it is compliant with the standard. Further, we extrapolate the analysis to other DLs.

2.2.1 Quality Model of EducaNext

Quality Factors and Criteria

The Effectiveness Model portrayed in Figure 3 is a form of quality model. It was developed by Simon [31] based on the results of a survey conducted with some members of Faculty of Economics in four European universities. The quality factor at the highest level is `Effectiveness`, which is the major yardstick for assessing whether the Portal can attain its ultimate goal, as reflected subjectively by the level of satisfaction that the users experience when using the Portal and objectively by the usage frequency. Further, the quality factor `Effectiveness` is related to two sets of quality factors subsumed by the two categories: *Brokerage Systems* and *Users*. As the quality factors under *Users* are beyond the control of software developers we are not going to elaborate them.

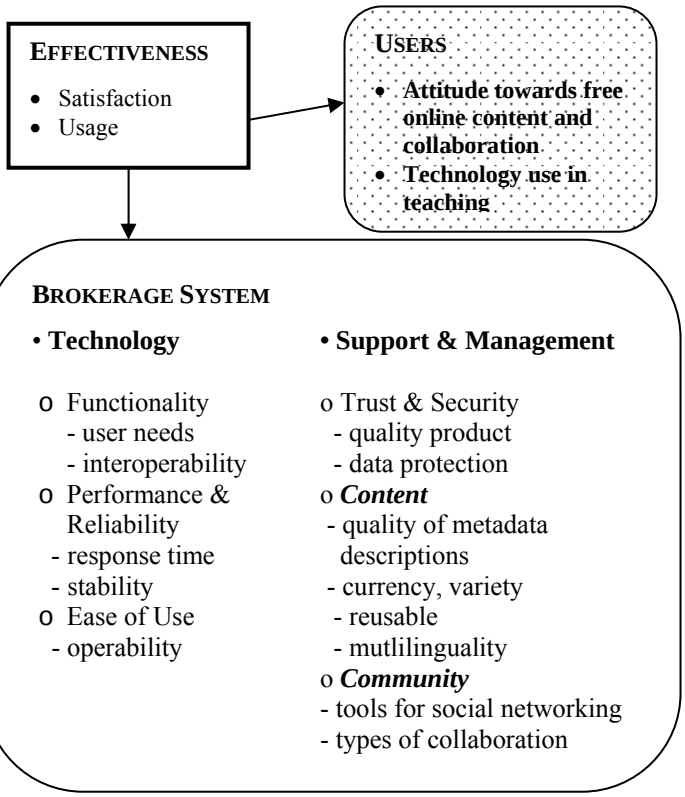


Figure 3: The Effectiveness Model of EducaNext (adapted from [31]);

Brokerage Systems: The quality factor `Functionality` refers to the features that are currently available and those that will be built into the Portal contingent on users' emerging needs. The quality factor `Performance & Reliability` refers to the general response time for queries being submitted to the system and to the stability and consistency of the system's behaviour. The quality factor `Ease of Use` denotes how simple it is as perceived by users to operate the system. The quality factor `Trust & Security` refers to the general image and reputation of the organization as perceived by users for delivering quality products and services, and to the policy for protecting intellectual property rights and personal data. Indeed, the issue of credibility and security are becoming more critical in the increasingly popular web-based transactions [4, 9, 10], for example, a proprietary DL for private commercial companies. The quality factor `Content` is actually composite, subsuming a set of interrelated attributes influencing the quality of learning objects offered in the Portal.

Quality Metrics and Measurements

Each of the criteria described above was translated into 86 open- and close-end questions of user survey, which was conducted in March 2004 to evaluate the Effectiveness Model. Furthermore, heuristic evaluation and think-aloud user tests have been performed to address the quality factor `Technology`.

Mapping to the ISO/IEC 9126-1 Standard

We have evaluated the compliance of the EducaNext Effectiveness Model with ISO 9126-1 by identifying so-called *mapped* and *extra* quality factors, which are and are not addressed in the standard, respectively.

Three mapped *External* quality factors (i.e., `functionality`, `reliability`, and `usability`) and three mapped *Quality-in-Use* factors (i.e., `effectiveness`, `safety` and `satisfaction`) were identified. Three extra quality factors, namely `Trust` `Content`, and `Community`, were also identified. Note that we have not (yet) delved so deep as to map the fine-grained metrics defined in 9126-2: External Metrics, 9126-3: Internal Metrics, or 9126-4: Quality-in-Use Metrics. In fact, some criticisms have been charged against the lengthiness and meaninglessness of a number of the metrics therein. Specifically, it is not clear what is being measured and what is used as reference for measurements. Consequently, it is difficult to make improvement recommendations based on the outcome of the metrics[3]. For instance, as noted in 9126-2, usability metrics measure the extent to which the software can be understood, learned, operated, attractive, and compliant with usability regulations and guidelines; the `Attractiveness` metric is calculated as the number of types of interface elements that can be customized divided by the number of types of interface elements. The higher

[3] Similar critiques have been charged against IEEE 1601 (1998) "Standard on Software Quality Metrics Methodology" [18].

the value the more attractive the interface is, implying that complete customization is most desirable. However, the assumption is highly questionable, especially when the problems of consistency, maintainability and costs are involved [29].

2.2.2 Quality Models of Other Digital Libraries

The Effectiveness Model of EducaNext can well exemplify quality models of other DLs. We look into three different non-European based DLs, namely **MERLOT** of the USA[4], **eduSource** of Canada[5], **EdNA Online** of Australia[6]. Similar to EducaNext, they are compliant with ISO 9126-1 by addressing `Functionality (Interoperability)`, `Usability` and `Reliability` as external quality factors. The two extra quality factors `Content` and `Community` are commonly addressed by these DLs as well.

For content quality control, both MERLOT and eduSource adopt a sophisticated peer review system. The three basic evaluation criteria are quality of content, potential effectiveness as a teaching tool, and ease of use[7]. In addition, eduSource has developed a set of criteria for evaluating quality of learning objects, such as interaction usability, accessibility, and reusability. Similar to EducaNext, EdNA Online puts emphasis on metadata quality, currency and variety of learning objects, and multi-linguality.

Given that most users of DLs are knowledge workers for whom knowledge-building community is a significant channel for them to share expertise and material, the quality factor `Community` is deemed essential. MERLOT communities and EdNA Online communities are built on disciplines and educational sectors, respectively, whereas eduSource communities, like EducaNext, are thematic, being defined by users themselves. Further, the three DLs address the quality factor `Accessibility` and emphasize compliance with the related guidelines and standards (e.g. W3C-WAI Web Content Accessibility). Besides, the three DLs adopt user-centred design approach by involving users in all stages of development.

To sum up, ISO 9126-1 is primarily concerned with qualities of software systems, which serve a vehicle or medium to convey or store contents, be they of mono- or multi-modality. Obviously, the quality of the vehicle does not necessarily relate to the quality of the content it carries. As a given standard cannot be all-encompassing to include everything, it is understandable that attributes pertinent to content quality controls are not addressed in ISO 9126-1. However, this quality factor is extensively addressed in ISO/IEC 19796-1 (see below). The quality factor `Community` addresses interactions between users. The question concerned is: How can a system enable user interactions that are essential for community building? The attribute `Interactivity` needs to be introduced under the quality factor `Usability` and be specified with reference to supporting features required for effective communication. On the other hand, the quality factor `Accessibility` by itself is so complex as to call forth a separate set of guidelines.

2.3 ISO/IEC 19796-1 (2005)

The final committee draft (FCD) of this standard was released in February 2005. It is especially relevant to DLs, as it addresses the quality factor `Content` rather extensively. Aligning with the conceptual model of existing DLs, peer review is deployed as the main mechanism for quality control. Besides, this standard explicitly addresses the issue of metadata quality – a core concern in the library science. Of particular interest is the framework for metadata creation, which is built upon Svenonius's [35] Principles of Bibliographic Description and Access: *Principle of User Convenience, Common Usage, Representation, Accuracy, Sufficiency and Necessity, Standardization and Integration*. These principles are philosophically and academically grounded and highly applicable to evaluating the catalogue of a DL and to addressing the quality factor `Content` (cf. Figure 3). As stated in the standard, there should be some fundamental principles that form the foundation of cataloguing rules. However, these principles have never been clearly listed in the rules [cf. 5]. Svenonius's work can somehow rekindle the research interest in this neglected topic. Nonetheless, meaningful metrics for assessing the compliance with these principles have not yet been available. This is a challenge facing DL designers, Information Science professionals and alike.

Further, ISO 19796-1 addresses the attribute `Collaboration` that is somewhat related to the quality factor `Community` mentioned above (Section 2.2.2). Specifically, `collaboration`, together with other associated attributes such as `communication`, `interaction`, and `experience exchange`, is mapped to the category `Responsiveness` of CELTSC[8]. The `Responsiveness` is measured in terms of average reply time to requests of different actors involved, and,

[4] http://www.merlot.org/Home.po
[5] http://edusource.licef.teluq.uquebec.ca/ese/en/overview.htm
[6] http://www.edna.edu.au/edna/go/pid/1
[7] http://taste.merlot.org/catalog/peer_review/eval_criteria.htm

[8] Chinese E-Learning Technology Standard Committee (http://media.cs.tsinghua.edu.cn/~pervasive/projects/e-learning/celtsc.html)

more interestingly, average 'complaints by student' as well as 'complaints per course'. These quantitative measures are apparently inadequate, because the quality of reply and underlying reasons of complaints are more relevant. Furthermore, caution needs to be exercised when borrowing concepts across cultures, i.e. the Asian standards may not be applicable to the Western contexts, and vice versa.

3. DISCUSSION

Digital Library, as a subclass of software application, should basically be bound to quality standards and models in Software Engineering and HCI. However, insuring the utility and usability of DLs is particularly challenging because of the vast heterogeneity of their users, the complexity of information organization, the compounding relationship between the intended and incidental content in the searching chain, and the huge volume of data. Do the prevailing software quality standards address the challenge? From the above analyses, though limited, the answer is ambivalent. Clearly, standards should be useful to a certain extent. Otherwise, they would have long been abolished [12]. Nonetheless, major weaknesses we identify in the software quality standards, at least those we have mentioned above, are the definitional issues such as the fuzziness of the key notion 'goal', and the practical issues such as inadequate estimation of usability metrics such as `Effectiveness` in 9241-11 and `Attractiveness` in 9126-2, and inconclusive relationships between usability metrics. Implications from the current review are that we need to substantiate the notion 'goal' and to specify not only quantitative but also qualitative usability metrics.

Furthermore, a quality model is to make the general term "quality" specific and useful when engineering requirements and to understand, control and improve a product [8]. *Should the adequacy of a quality model be evaluated based on its coverage of the widely adopted ISO/IEC 9126-1 standard?* The answer, unfortunately, is uncertain. On the one hand, assuming that ISO 9126-1 is a generic basic framework applicable to all software products, a quality model will be seen as defective if it omits any of the characteristics addressed in the standard. On the other hand, this generic standard needs to be customized to address specific organizational constraints and product goals; it is legitimate to select a subset of the characteristics without jeopardizing the adequacy of the model. Conversely, *should the adequacy of ISO/IEC 9126-1 be called in question when it does not address extra quality factors highly relevant to a DL*, such as `accessibility`, `trust/security`, `user convenience` and `community/collaboration`? This answer tends to be negative. In fact, specific standards and guidelines have been evolving to address emerging quality issues engendered by new IT paradigms, such as W3C Web Content Accessibility Guidelines, and ISO 17799 Information Security Standards. Clearly, it is impractical to sweep all quality issues under a single standard.

4. CONCLUDING REMARK

As this study is a work in progress, no definitive conclusion pertinent to the important research question raised in the beginning can be drawn. Nonetheless, it is clear that compliance with a single standard is far from being sufficient to address key features of DLs. Instead, so-called pluralistic conformance with multiple standards is deemed necessary. We are extending the current empirical and analytical work to garner more data about the DLs covered in the foregoing text and about the emergent ones such as iLumina Digital Library (USA) and NIME (Japan). By comparing and contrasting their commonalities and differences in design, structure and socio-cultural application context, insights into developing a generic model of DL and further improving the related standards can be gained. Besides, we shall examine other relevant standards such as ISO/IEC 14598 [15] and ISO/IEC 16982 [16]. Furthermore, systematic empirical analysis of how DLs are actually deployed in real contexts is indispensable as it will shed some light on the problem of how to develop a useful and usable DL.

REFERENCES

1. Baeza-Yates, R., & Ribeiro-Neto, B. (1999). *Modern information retrieval*. New York: ACM Press.
2. Blandford, A., Keith, S., Connell, I., & Edwards, H. (2004). Analytical usability evaluation of digital libraries: a case study. In Proceedings *of Joint Conference of Digital Libraries (JCDL'04)*, June 7-11, 2004, Tucson, Arizona, USA.
3. Blandford, A. (in press). Interacting with information resources: Digital libraries for education. To appear in *International Journal of Learning Technology*.
4. Butler, M., Leuschel, M., Presti, S.L., & Turner, P. (2004). The use of formal methods in the analyses of trust (Position paper). In *Proceedings iTrust 2004* (LNCS 2995, pp. 333-339)
5. Duval, E., Hodgins, W., Sutton S., Wiebel, S.L. (2002). Metadata principles and practicalities. *D-Lib Magazine, 8*(4), http://www.dlib.org/dlib/april02/
6. Elliot, M., & Kling, R. (1997). Organizational usability of digital libraries. *Journal of the American Society for Information Science, 48*(11), 1023-1025.
7. Fenton, N.E., & Pfleeger, S. (1997). *Software metrics* (2nd Ed.).
8. Firesmith, D. (2003). Using quality models to engineer quality requirements. *Journal of Object Technology, Vol. 2, No. 5*, pp. 67-75

9. Fogg, B. J., et al. (2001).What makes web site credible? A report on a large quantitative study. In *Proceedings CHI 2001*.
10. Flechais, I., Sasse, A., Hailes S. M.V. (2003). Bringing security home. In Proceedings of *New Security Paradigms Workshop* (NSPW'03), Switzerland. ACM/SIGSAC.
11. Frøkjær, E., Hertzum, M., and Hornbæk, K. (2000). Measuring usability: Are effectiveness, efficiency and satisfaction really correlated? In *Proceedings of CHI* 2000, 1-6 April, the Hague, Netherlands, pp. 345-352.
12. Good, R. (2003). *Standards: Do we really need them?* Online at: http://www.masternewmedia.org/2003/12/26/ standards_do_we_really_need.htm
13. ISO/IEC 9241-11 (1996). *Guidance on Usability, Ergonomic Requirements for Office Work with Visual Display Terminals*
14. ISO/IEC 9126-1 (2001) *Software Engineering – Product Quality – Part 1: Quality Mode*l
15. ISO/IEC 14598 (1998). *Information technology -- Software product evaluation*
16. ISO/IEC 16982 (2002) Ergonomics of human-system interaction -- Usability methods supporting human-centred design
17. ISO/IEC 19796 (2005). *Information technology -- Learning, education and training -- Quality management, assurance and metrics -- Part 1: General approach*
18. Kaner, C. (2004). Software engineering metrics: What do they measure and how do we know? In *Proceedings of 10th International Software Metrics Symposium.*
19. Kelly, D., & Cool, C. (2002). The effects of topic familiarity on information search behaviour. In Proceedings of the Second JCDL, pp. 74-75.
20. Kissel, G.V. (1995) The effect of computer experience on subjective and objective software usability measures. In *Proceedings of CHI' 95* (New York: ACM Press), pp. 284-285
21. Klobucar, T. (2005). *Search performance evaluation*. Deliverable D5.3 of the ELENA Consortium.
22. Lave, J. (1988). Cognition in practice. Cambridge: Cambridge University Press.
23. Lewis, J. R. (1995). IBM computer usability satisfaction questionnaires: Psychometric evaluation and instructions for use. *International Journal of Human-Computer Interaction, 7*, 1, 57-78.
24. McCall, J., Richards, P., & Walters, G. (1977). *Factors in Software Quality. Vol. 1, 2, & 3.*
25. Moody, D. L., Sindre, G., Brasethvik, T., and Sølvberg, A. (2003). Evaluating the quality of information models. In *Proceedings of International Conference Software Engineering (ICSE) 2003*, May 3-10 2003, Portland, Oregon, pp. 295-307.
26. Nardi, B. (1996), Studying context : a comparison of activity theory, situated action models and distributed cognition. In B. Nardi (Ed.), *Context and consciousness: activity theory and human-computer interaction* (pp. 69-102). MIT, MA.
27. Nielsen J. and Levy, J. (1994). Measuring usability: Preference vs. performance. *Communications of the ACM*, 37 (4), pp. 66-75.
28. Sandusky, R. J. (2002). Digital library attributes: Framing usability research. In *Proceedings of JCDL'02 Workshop on Usability of Digital Libraries* (http://www.uclic.ucl.ac.uk/annb/DLUsability/JCDL02 .html)
29. Schmitt, S., & Bergmann, R. (1999). Applying case-based reasoning technology for product selection and customization in electronic commerce environments. In: *Proceedings* of *12th International Bled Electronic Commerce Conference*
30. Schulte, M., & Huber, O. (2003). Information search in the lab and on the Web. *Behaviour Research Methods, Instruments & Computers, 35*(2), 227-235.
31. Simon, B. (2001). E-Learning an Hochschulen: Erfolgsfaktoren und Gestaltungsräume von Wissensmedien. Eul-Verlage, Lohmar, Köln.
32. Stutt, A., & Motta, E. (2004). Semantic learning webs. Journal of Interactive Media in Education, 10. Online: http://www.jime.open.ac.uk/2004/10
33. Suchman, L. (1987). *Plans and situated actions*. Cambridge: Cambridge University Press
34. Sumner, T., & Marlino, M. (2004). Digital libraries and educational practice: A case study for new models. In *Proceedings of Joint Conference on Digital Libraries* (JCDL'04), pp. 170-178.
35. Svenonius, E. (2000). *Intellectual foundation of information organization*. Cambridge, Mass: MIT Press.
36. Yeo, A.W. (2001). Global-software development lifecycle: An exploratory study. In *Proceedings of CHI'01* (pp. 104-111). New York: ACM Press.

User Interface Quality and Context of Use

Asbjørn Følstad
SINTEF ICT
0314 Oslo, Norway
asf@sintef.no

ABSTRACT
Context of use should be an important aspect of user interface quality models (UIQM). For a range of applications, the complexity of their context of use suggests that it is not feasible to include low level context of use specifications in a UIQM. It is suggested that context of use in a UIQM should include (1) high level specifications of prioritized contexts of use, (2) descriptions of experts possessing the context knowledge needed to specify context-dependent attributes of a quality model, and (3) processes for these experts to discuss and define context-dependent attributes. It is further suggested that context of use may be viewed as consisting of two attributes: domains and ICT environment, where a domain includes the tasks, goals, and organisational environment of one or more user groups, and ICT environment includes hardware, software, and network. It may be relevant to allow the inclusion of both domain experts and experts on ICT environment in a UIQM. The argument of the paper is supported by examples from UI development and evaluation of two kinds of applications: mobile applications and solutions for customer guidance in e-commerce.

Author Keywords
User Interface Quality Model, Context of Use

THE ROLE OF CONTEXT OF USE IN USER-CENTRED DESIGN (UCD) AND SOFTWARE QUALITY MODELS

The importance of an application's context of use is well established within the tradition of User-Centred Design (UCD), as it is in the field of Human-Computer Interaction (HCI) in general. This may be exemplified by reference to the international standard for Human-centred design processes for interactive systems, ISO 13407 [1], where the first part of the design process is to "understand and specify the context of use"; or the international standard ISO 9241-11 on ergonomic requirements for office work with visual display terminals - guidance on usability [2], where usability measures are applied to the outcome of the interaction between the product to be evaluated and its context of use.

ISO 9241-11 (and subsequently ISO 13407) defines context of use as "users, tasks, equipment (hardware, software and materials), and the physical and social environment in which a product is used". In the same standard, an example description of an application's context of use may consist of "users" described along 19 dimensions, "tasks" described along 10 dimensions, "equipment" described along 8 dimensions, and "environment" described along 32 dimensions. As may be understood from this example, the concept of context of use in UCD is truly broad. Also, the complexity of a low level context specification for almost any given application for user interaction will quite easily become overwhelming.

The international standard for software product quality, ISO 9126 [3], describes a two-part model for software product quality, focusing on (1) internal and external quality and (2) quality in use.

External and internal quality is decomposed in six attributes, one of which is usability. The standard's recommended methods to apply usability metrics for external quality include user test or tests of the product in use (see ISO 9126-2). Context of use is not explicitly referred to in association with external quality, but is included implicitly through the use of user tests as metric for external usability quality.

Quality in use is decomposed into four attributes: Effectiveness, productivity, safety, and satisfaction. All the four attributes of quality in use are defined with reference to "a specified context of use"; the proposed metrics should measure "the extent to which a product meets the needs of specified users to achieve specified goals with effectiveness, productivity, safety, and satisfaction in a specified context of use". Further, it is stated that "[e]valuating quality in use validates software product quality in specific user-task scenarios" (see ISO 9126-4).

Both with regard to the external quality attribute of usability and the attributes of quality of use, context of use serves as a qualifier; qualifying either the sub-characteristics of usability (e.g. understandability and learnability) or the quality in use attributes of effectiveness, productivity, safety, and satisfaction.

THE ROLE OF CONTEXT OF USE IN A USER INTERFACE QUALITY MODEL (UIQM)

Context of use as defined and utilized within the field of UCD (e.g. in ISO 9241-11) has been incorporated in the international standard for software quality. Similarly, context of use should be included in a User Interface Quality Model (UIQM), in particular if the UIQM is to be regarded as a software quality model.

I trust that the idea of including context of use in a UIQM is considered quite uncontroversial. Context of use is probably accepted by default as one of the core aspects of

both a UIQM and associated user interface (UI) quality metrics.

However, the way context of use is to be included in a UIQM needs to be considered carefully. It may be argued that context of use, as defined and exemplified in ISO 9241-11, may be too broad to be of great help in systematic development of requirements specifications and evaluation procedures for UIs. When context of use is understood as practically everything outside the application that may have an impact on its usability or quality in use, it seems reasonable to assume that any detailed specification of context of use will easily suffer from being incomprehensive. Also, in a detailed specification of context of use, it may be difficult to separate the critical aspects of the applications' context of use from the less important aspects.

In the present paper I will suggest an inclusion of context of use in a UIQM, where context of use is understood as consisting of the attributes of "domains" and "ICT environment". A domain is taken to include one or more user groups and their tasks, organizational environment and physical environment. The ICT environment is understood as the application's associated hardware, software, networks and support systems. I will further suggest that domain specifications should only be made on a high level, and that attributes of a UIQM referring to domain specifications should be specified and measured in association with domain experts, rather than with reference to low-level specifications of context of use.

It should be noted that my suggestion to specify certain quality attributes in association with experts, does not imply the introduction of process quality measures in the UIQM. I do not suggest that UI quality explicitly should be evaluated with regard to the underlying process of e.g. developing a sufficient context of use specification. Rather, my suggestion on a general level implies that a UIQM should include procedures for how some product quality attributes should be specified.

To support my argument I will exemplify the relationship of context of use and UI quality for two kinds of applications: mobile ICT applications and applications for customer guidance on web-sites. Following this I will elaborate on how context of use may be included in a UIQM and the implication this may have for UI quality metrics and evaluation.

CONTEXT OF USE AND UI QUALITY OF MOBILE ICT

Evaluation of mobile UIs quickly makes the evaluator aware of the tremendous impact of context of use in the quality of a mobile UI. The context of use for a mobile UI may be varied and unpredictable, both with regard to the user's social situation, physical conditions (lighting e.g.), connection to other technological devices, and network facilities [4]. Further, a mobile service may typically be an assistive technology, supporting the user in reaching her goal without being the goal itself. It is important to be aware of the potential competition between the mobile service and its environment. Mobile services should be available when needed without being overly intrusive, and not disrupt the user's natural workflow [5].

In the research projects Mobidist (now finished) and Umbra (ongoing), we have evaluated mobile applications for e.g. newspaper deliverers, parking wardens, and medical personnel in hospitals. Some of the results from these evaluations give insights in how context of use may impact UI quality.

The impact of the ICT environment was made evident in the UI quality evaluation of a mobile application for newspaper deliverers (see [6]). This application was developed through a User-Centred process, and the field evaluations that were conducted showed that the application itself represented no critical user problems. However, the same field evaluations showed that the standard software and hardware of the handheld terminal used for the first running prototype represented critical user problems. Examples of this included users failing to understand how network connectivity status was communicated, and not being able to regulate the back-light of the device's screen (!).

The impact of an application's domain on UI quality judgments was made equally evident in two evaluations of mobile applications for parking wardens and medical personnel at a hospital [7]. The evaluations were conducted as cognitive walkthrough in groups adapted for evaluation of mobile ICT. For each application two evaluations were conducted; one with domain experts (parking wards and nurses respectively) and one with usability experts (professionals on UI development and evaluation). The evaluation results showed that the usability problems and design suggestions identified by the domain experts were prioritized significantly higher by the developers, and that the feed-back from the domain experts were appreciated by the developers as more relevant and useful.

The lesson that may be learnt from the above evaluations is twofold. First, the impact of the ICT environment (and also the domain) on UI quality may be both surprising and associated with minor details in the total context of use of the applications; suggesting that it may be difficult to secure high UI quality through a sufficiently detailed specification of context of use. Second, domain experts may serve as a valuable source for direct feedback on UI quality, even though the domain knowledge represented by the domain expert largely remains tacit knowledge.

CONTEXT OF USE AND UI QUALITY FOR CUSTOMER GUIDANCE IN E-COMMERCE

Development of customer guidance for e-commerce is a relatively new area of interest where the impact of context of use of UI quality is highly felt, both with regard to specification and evaluation. Customer guidance includes aspects of e-commerce aimed at informing and helping the customer, as well as providing the customer with new insight and ideas. E.g. an Internet store selling mobile phones may provide customer guidance with regard to (1) assistance in choosing the phone best suited to the customer's needs, (2) instructions on how to use the advanced functionality of the phone, and (3) ideas and inspiration on how to start using the phone in new ways.

In the ongoing research project "eHandel gir mer!" where the aim is to develop methods for identifying requirements and evaluating web UIs for customer guidance, it has become increasingly evident that the context of use for customer guidance UIs is both highly important and highly complex. Users of customer guidance may hold a wide range of goals, quite possibly goals that they are only able to articulate on a very high level (like in "I am in serious need of help to get my new phone to work properly"). Also, the completion of goals (e.g. getting the phone to work properly) may not be correlated with the ease of use of the customer guidance. It may for example be self-evident how to click though the customer guidance, even though the guidance does not solve the user's problem or have the desired effect on the user's skill level. Further, the customer guidance on a particular web-site needs to function together with other aids in the vicinity of the user (e.g. other web resources, paper-based instructions, off-line training courses or the company help-desk).

Early work in the project "eHandel gir mer!" indicates that the relevant context of use for customer guidance is complex to the extent that it is not possible to describe in a detailed specification. In addition, the context of use may be rapidly changing, e.g. as the product line of the e-commerce site changes. On the bright side, the e-commerce organizations will most likely have employees with strong insight in customer's needs with regard to customer guidance, e.g. based on direct customer contact through first-line customer service. These employees may be considered domain experts on customer guidance within their company's area of interest (e.g. mobile phones), and may be successfully utilized in the specification, development and maintenance of customer guidance functionality for their e-commerce web site. The domain knowledge of these personnel resources may not be explicated through detailed specifications of context of use. Even so, the inclusion of domain experts provides the developer with both a source to requirements generated through context of use, as well as a resource for design feedback and evaluation.

CONTEXT OF USE, UIQMS AND UI QUALITY METRICS

As may be seen from the examples above, context of use may have a strong impact on UI quality. The examples also show that the UCD approach to context of use as something to be specified to the level that the developer and/or usability expert can base a requirements specification on it, may not be a viable approach to context of use as an element in a UIQM.

The detailed understanding of context of use often only resides as tacit knowledge in the experience base of experts. Above, I suggested an analysis of context of use in two components: ICT environment and domain. Following this argument, understanding context of use requires the utilization of the tacit knowledge of domain experts, and possibly also the tacit knowledge of experts on the ICT environment of the applications. Examples of domain experts may in principle include end-user representatives selected on the basis of some additional criteria (e.g. work experience and ICT competency) or persons experienced with the domain by other means (e.g. through work with end-user support).

How then should context of use be included in UIQMs? I suggest that a specification of context of use to be used in a UIQM may include three elements:

1. A listing of prioritized domains and ICT environment with associated high-level description
2. A listing of required domain and/or ICT environment experts to cover the prioritized domains and ICT environment, each with specifications of the characteristics by which to identify each expert category
3. Reference to defined procedures for involving experts for detailed input on context of use, to be used when specifying context-dependent quality attributes.

My suggestion on a general level implies that a UIQM should include process descriptions on how at least some product quality attributes should be specified. However, as noted above, this should not be confused with a suggestion to include process quality attributes as part of a UIQM. The process suggested is only meant to support the specification of product quality attributes.

An example of this approach to context of use in UIQM may be to specify the context of use of a customer guidance application. The context of use for such a guidance application may be specified as (1) a high level description of the domain of customer support for consumer electronics for a particular company, with (2) the domain experts specified as customer support personnel of given seniority and experience, and finally indicating (3) a procedure for domain expert involvement for defining the relevant quality attributes. Such a procedure could include one or more workshops with domain experts in early design, using rapid prototypes as a basis to provide detailed specifications of one or more of the general quality attributes of effectiveness, productivity, safety and satisfaction.

It should be noted that in this example the specification of context of use is not in itself detailed enough to support specification of quality attributes. Still, the requirements developed regarding a given quality attribute is precisely specified through the involvement of domain experts.

By the same token, quality metrics may be specified for any quality attribute depending on context of use by the involvement of experts on context of use (either domain experts or experts on ICT environment). The specification of quality metrics, such as the specification of quality attributes, will not depend on a detailed specification of context but on the involvement of experts in metrics development.

CONCLUSION

A suggestion on how to include context of use in UIQMs has been presented. The suggestion is based on the dual assumption that detailed knowledge on context of use is necessary in order to specify and use a range of quality

attributes and quality metrics, and that this knowledge may not be successfully specified in sufficient detail. The proposed solution to the challenge implied in the assumptions, is to make context of use specifications include descriptions of experts as well as references to procedures for involving the experts in the specification of context-sensitive quality attributes and associated metrics.

ACKNOWLEDGEMENTS

The present paper was written as part of the research projects UMBRA (Development of mobile user interfaces) and "E-handel gir mer!" (Design and evaluation of customer guidance in e-commerce). UMBRA runs from 2004 to 2005 and is partly financed by the Norwegian Research Council (NRC) through the IKT-program. "E-handel gir mer!" runs from 2005 to 2006 and is financially supported by the NRC through the Puls-program.

REFERENCES

1. ISO 13407 (1999). Human-centred design processes for interactive systems
2. ISO 9241-11 (1994). Ergonomic requirements for office work with visual display terminals- guidance on usability
3. ISO 9126 (2000) Information technology – Software product quality
4. Nilsson, E. G., & Rahlff, O. W. (2003). Mobile and Stationary User Interfaces, Proceedings of HCI International 2003
5. Consolvo, S., Arnstein, L., & Franza, B. R. (2002). User Study Techniques in the Design and Evaluation of a Ubicomp Environment. UbiComp 2002
6. Følstad, A., Rahlff, O.-W. (2005) Challenges in Conducting User-Centred Evaluations of Mobile Services, Proceedings of HCI International 2005 (in press)
7. Følstad, A. (2005) Analytisk usability-evaluering i gruppe, SINTEF project report

A domain-oriented approach in structuring user interface guidelines

Costin Pribeanu
ICI Bucureşti, Romania
E-mail: pribeanu@ici.ro

ABSTRACT

User interface guidelines are organized in guideline bases and sections according to various criteria. Accessing guidelines for a target interface could be done either by following guidelines organization or by using key words. This paper aims at presenting an alternative approach in structuring user interface guidelines. In this respect, we propose a layered structure having on top domain-oriented criteria and heuristics.

Categories and Subject Descriptors:

D.2.8 [**Software Engineering**]: Metrics, H.1.2 [Human-Machine Systems]: Human factors H.5.2 [**Information Interfaces and Presentation**]: User Interfaces – evaluation, standardization, user centered design

Keywords:

Guidelines, heuristics, domain-oriented criteria, usability

INTRODUCTION

User interface design is based on a substantial body of knowledge from ergonomics, psychology, software engineering and other disciplines. This design knowledge could be captured in a prescriptive form as design principles to be followed by designers of interactive systems. A wide range of design principles have been proposed, such as: ergonomic criteria, design heuristics and guidelines.

Bastien and Scapin [1] proposed a set ergonomic criteria consisting in 18 elementary design and evaluation principles grouped into 8 categories. For each ergonomic criterion the prescription is providing with definition, rationale, comments, and examples of guidelines. Ergonomic criteria are used for several purposes: describing the rationale for using guidelines, evaluating the ergonomic quality of user interfaces, indexing, and accessing guideline databases.

Molich & Nielsen [3] proposed heuristic evaluation as an inspection method focusing on widely recognized principles. Heuristics are principles that enable the evaluator to identify most of the usability problems However, a small set of principles require skilled experts in order to bring significant results and might be subjective.

Reducing the inherent subjectivity of inspection-based methods is possible by expanding the reference base. Guidelines are providing with more detailed prescriptions, following a guideline template which includes, among others: a statement, a list of bibliographic references, a rationale justifying the guideline, positive and negative examples and a set of relationships established with other guidelines. A general guideline model proposed by Vanderdonckt [7] also includes other entries, such as: ergonomic criteria respected, utility and usability factors satisfied by the guideline.

Usually, guidelines are organized in guideline bases and sections according to various criteria. Accessing guidelines for a target interface could be done either by following guidelines organization or by using key words. A problem with existing collections is that they are heterogeneous as regarding the structure and the number of attributes.

This paper aims at presenting a domain-oriented approach in structuring user interface guidelines. User interface guidelines should satisfy both ergonomic and specific requirements. In this respect domain-specific principles that have been effective in the target activity domain are sinthetized in design heuristics. Their role is to contextualize the guidelines in the design and evaluation process.

The rest of this paper is organized as follows. In the next section we will discuss the change of focus from usability to quality in use in two ISO standards. Then, we will briefly discuss some issues regarding tools for working with guidelines. In section 4 we will present our approach by illustrating it with two examples of web design heuristics and guidelines targeting two application domains: online communities and e-commerce.

QUALITY AND USABILITY OF INTERACTIVE SYSTEMS

The design of user interfaces is part of the overall design of a software system. The evolution of HCI in the last decade shows a maturation of the discipline and a closer integration of usability evaluation methods in the iterative design process. This trend is mirrored someway by the evolution of two concepts which are relevant for user interface evaluation: usability and quality of software systems.

In the former version of ISO 9126:1991 standard, usability has been defined as a software quality attribute that bears on the capability of being easy to comprehend, understand, and operate.

Later on, the ISO standard 9241-11:1994 took a broader perspective on usability as the extent to which a product can be used by specified users to achieve specified goals effectively, efficiently and with satisfaction in a specified

context of use. In this standard, the context of use has four main components: user, tasks, platform and environment.

These definitions were revised and integrated in the new version of ISO 9126 standard on quality of software systems as follows:

- Usability is the capability of a software system to be understood, learned, used, and liked by the user when used under specified conditions. (ISO/IEC 9126-1:2001)
- Quality in use is the extent to which specified users accomplish specified goals with effectiveness, productivity, security and satisfaction in a given context of use. (ISO/IEC TR 9126-4:2004)

The quality in use is closely related with four quality characteristics: functionality, usability, reliability and efficiency. As such, it is more comprehensive and requires a broader view on design and evaluation of interactive systems targeting a specified context of use.

GUIDELINE BASED EVALUATION

Guidelines are widely used for the design and evaluation of web applications. Although the interest for respecting guidelines is increasing, guidelines compliance of web sites is still low. Based on the analysis of 15 major e-commerce sites, Nielsen reports that on average, the sites followed 49% of the e-commerce usability guidelines in 2001 [4].

Guidelines that are covering almost all known usability problems could heavily improve the usability of an interactive system. Unfortunately, the huge amount of existing guidelines (and the process of collecting them is ongoing) makes very laborious the evaluation process.

Automate methods are using a software program to perform the usability evaluation. This kind of tools will work with guidelines in that they will use guidelines as a reference base for evaluation and will enable systematic usability checking that is not possible to be done by an expert in a reasonable time. Vanderdonckt [7] identified five development milestones towards a tool for working with guidelines:

1. guidelines collecting – gathering guidelines from available ergonomic sources;
2. guidelines organization - sorting and classifying them into a single organizing framework;
3. incorporation guidelines into approach - developing a methodology for finding and applying relevant guidelines to a particular field of interest;
4. guidelines operationalization - giving a computational representation of the structured guidelines set for manipulation by computer based tools;
5. guideline use for design or evaluation

This approach is a gradual shifting from guideline based evaluation to automated testing of web sites. The core milestone is the third one since it makes possible to select relevant guidelines from a broad guidelines corpus and it heavily depends on guidelines organization.

Structuring of usability prescriptions is not a new problem. Several approaches have been motivated by the evolution of usability concept. Seffah, Kecci and Donayee [8] proposed a layered model of usability comprising factors, criteria, metrics and data. The relation between factors and criteria is most important although it is limited to usability components in ISO 9241-11 and their associated attributes.

DOMAIN-SPECIFIC CRITERIA

In our approach, a factor is a criteria type: a common denominator for a group of related criteria. There are several factors influencing the evaluation criteria and design principles for a target application domain but their relevance is varying according to the context of use.

Our framework for structuring design guidelines is depicted in Figure 1.

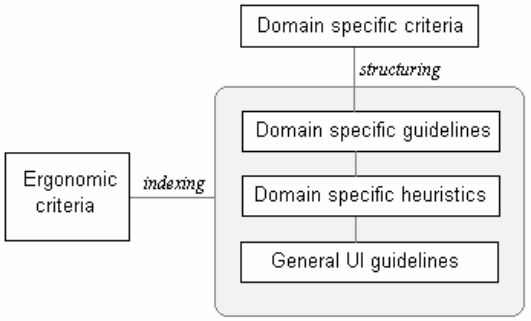


Figure 1. A layered structure of usability prescriptions

Each application domain has its own values and priorities which are driving the activity. In turn, these values could be embodied in design principles or heuristics and further expanded in more detailed guidelines. According to the application domain, we might need domain-specific criteria related to marketing, commercial, social, or educational aspects.

A problem with domain specific criteria is that is difficult to collect domain-specific heuristics and guidelines. Most valuable design heuristics are internalized by experts in the domain area and rarely made available in public collections. Therefore an interdisciplinary approach is mandatory in order to collect a useful and well structured collection of guidelines.

On another hand, ergonomic criteria are applying for user interfaces targeting any application domain. In this respect, the ergonomic factor is a general factor in structuring design guidelines. However, in our approach ergonomic criteria are mainly targeting general user interface guidelines.

Although domain-specific heuristics are targeting an application domain they are not limited to one factor. For

example, a social criterion like trust applies for both online communities and e-commerce web sites.

Heuristics could also take a general guideline model, including statement, examples, and references. This way moving from heuristics to guidelines is a seamless transition. Although the guideline and / or heuristics format is beyond the scope of this paper, we think that adopting a similar format, could lead to more consistent collections.

Specific criteria for on-line communities

Web sites for on-line communities have to respect social criteria. A set of criteria proposed by different approaches to building on-line communities is given below [2, 5]:

- Clear definition of group boundaries in order to prevent people to take an opportunistic and selfish approach by using common resources without contribution;
- Rules and institutions (rules to use the collective goods, changing of rules and respect paid by authorities to the right of devising rules);
- Monitoring and sanctioning (monitoring by the members of the community and a graduated system of sanctions).

In her book, Preece [6] proposed a working definition for on-line communities, which consists in people, a shared purpose, policies (assumptions, rituals, protocols and rules) and computer systems. The focus in this definition is on sociability rather than usability. As it could be observed, this working definition integrates most of design principles mentioned above. She also gave a name to these specific criteria: sociability, as being concerned with "planning and developing social policies which are understandable and acceptable to members, to support the community's purpose".

In order start and develop on-line communities it is also important to attract people and to keep them as active participants. Since the virtual space provided by a web site is different from the traditional public space it is not an easy task to convince people to join.

Based on the principles described above, in a previous work a set of heuristics has been proposed that could be appropriate for the design, and evaluation of web sites supporting on-line communities [7]. The set is presented in Table 1.

We assume that the purpose of the community is the social interaction between the inhabitants of the locality. In this respect they could apply for membership. First four heuristics on the first level are mainly concerning the public part of the web site.

1. Provide with means to attract people
   - 1.1. Connect people to the social life in the community.
     - 1.1.1. Acknowledge and comment important events
     - 1.1.2. Provide with links to people in the community
   - 1.2. Create an attractive social space.
   - 1.3. Find partners able to foster the social interaction.
   - 1.4. Record and present the collective actions carried on.
2. Ask people to identify themselves.
3. Provide with clear membership conditions.
4. Explain clearly the policies and rules of conduct.
5. Provide with shared assets and stimulate individual contributions
6. Provide with means for monitoring and sanctioning.
7. Find appropriate social moderators.
8. ...

Table 1. A sample set of sociability heuristics

Note that the principles in Table 1 are rather illustrative than prescriptive and include only a sample set of heuristics. Providing with a complete set of heuristics is beyond the purpose of this paper.

The next level is formed by web site design guidelines. Like heuristics, guidelines could be organized in a hierarchy for more general to more particular rules. A sample lower level guideline is given below.

Social criteria: Create an attractive social space.

Higher-level guideline: Decorate the web site to celebrate holidays and special occasions.

Statement: Remove all ornaments and decorations when the special occasion is over.

Rationale: Users will have the feeling the web site is out of date.

Source: Jakob Nielsen's Alertbox, October 28, 2002: Celebrating Holidays and Special Occasions on Websites

Table 2. A lower level guideline

The layered structure of the design knowledge together with a consistent definition at all levels provides with a better structuring of design guidelines.

General web design guidelines are beyond the domain-oriented guidelines since they are addressing general user interface design issues related to web-based applications, such as home page design and navigational links. Most ergonomic guidelines derived from ergonomic criteria are applying here.

Specific criteria for e-commerce

In a similar way, heuristics and guidelines for e-commerce could be structured by domain specific criteria that are corresponding to well established principles in marketing and commercial transactions.

In Table 3 a sample set of heuristics for e-commerce is provided.

1. Provide the client with needed information
   - 1.1. Provide contact information.
   - 1.2. Provide information about shipping.
   - 1.3. Specify the terms and conditions
   - 1.4. Record and present the collective actions carried on.
2. Provide with means to compare products
   - 2.1. Display products simultaneously
   - 2.2. Enable the user to choose from compared products
3. Provide navigation facilities
4. ...

Table 3. A sample set of e-commerce heuristics

Each heuristic is related to a specific e-commerce criterion (information, promotion, distribution, trust etc).

These heuristics could be further expanded into more detailed guidelines, like in Table 4.

Commercial criteria: Information.
Heuristics: Display products simultaneously to facilitate comparison.
Statement: Provide with appropriate means to support comparison
Rationale: Help users to compare different products in different contexts of using the web site
Ergonomic criteria: grouping / distinction
Lower level guidelines:
Provide with a table of features and specifications
Provide with pictures and textual descriptions
Source: IBM design Guidelines – E-Commerce

Table 4. An e-commerce design guideline

It is important to provide the designer / evaluator with both higher and the lower level entities that are associated with a guideline. This way, (s)he can perceive at a glance the guidelines organization and easy browse the guideline collection.

CONCLUSION AND FUTURE WORK

The quality of user interfaces is a trade-off between utility and usability. Utility is related to the appropriateness of computer tools for a given domain / problem. In this respect, evaluation should address the core issues in the specified area, looking for principles that were proven to produce successful designs.

In this paper we proposed a layered structure of guideline data bases that relates prescriptions to domain-specific criteria in order to support the quality in use via more context-oriented collections of design knowledge. This approach has several benefits:

- It integrates three types of prescriptions widely used in design and evaluation: criteria, heuristics and guidelines.
- It provides designers and evaluators with relevant heuristics and guidelines for a given application domain.
- Domain experts could be more effectively involved in the design and evaluation of interactive systems.

Guideline-based evaluation requires tools for working with guidelines. The domain-oriented organization of guidelines is challenging the design and implementation of tools for working with guidelines with at least two problems:

- Integration of existing general UI guidelines that are either derived from ergonomic criteria or are specific to an interface type (e.g. web).
- Integration of heuristics / guidelines which are specific to different but related domains. (e.g. education and entertainment criteria used in edutainment applications).

REFERENCES

1. Bastien, C.J.M. and Scapin, D., *Ergonomic Criteria for the Evaluation of Human Computer Interfaces*, Technical Report No.156, INRIA, 1993.
2. Kollock, Peter. 1996. "Design Principles for Online Communities." *Harvard Conference on the Internet and Society*. Also published in PC Update 15(5): 58-60. June 1998.
3. Nielsen, J., and Molich, R. "Heuristic evaluation of user interfaces". *Proc. ACM CHI'90 Conference*. (Seattle, WA, 1-5 April), 249-256. (1990).
4. Nielsen, J. (2002) Jakob Nielsen's Alertbox, June 24, 2002: Improving Usability Guideline Compliance. http://www.useit.com/alertbox/
5. Ostrom, Elinor. *Governing the Commons: The Evolution of Institutions for Collective Action.* New York: Cambridge University Press. 1990.
6. Preece, J. *Online Communities: Designing Usability, Supporting Sociability*. John Wiley & Sons. 2000
7. Pribeanu, C. "Towards a framework for the evaluation of web sites intended to support on-line communities". *Proc. of the COST 269 Conference*, (Helsinki, 3-5 September 2003). Medialab – UIAH, Helsinki, pp.72-75. (2003)
8. Seffah, A., Kecci, N., Donyaee, M. "QUIM – A Framework for Quantifying Usability Metrics in Software Quality Models". *Proceedings of APAQS'01*. Hong Kong. pp. 311-318. (2001)
9. Vanderdonckt, J. "Development milestones towards a tool for working with guidelines". *Interacting with Computers, Vol.12, No.2*, 1999. pp 81-118.

Factors affecting credibility of e-shops in Poland

Igor Garnik
Gdansk University of Technology
Faculty of Management and Economics, Ergonomics Dept.
Gdansk, Poland
e-mail: Igor.Garnik@zie.pg.gda.pl

ABSTRACT
This paper presents main problems concerning e-commerce development in Poland and in other countries of Eastern Europe, in particular, links between a consumer's trust and the cultural background in the region. The first results of a pilot study being a part of the author's research project are presented in the paper. As they show, factors affecting credibility of e-shops are strictly related to usability, reliability, functionality and efficiency of the websites – the crucial issues of user interface quality models.

Keywords
Internet, e-commerce, trust, credibility, user interface quality, usability

INTRODUCTION

In the last decade, dynamic development of the Internet has been strongly stimulating its use in business areas. However, business activity in the Internet still encounters many obstacles.

For example, the results of a research conducted by InterCommerce Corporation [3] showed that in the United States (being the country of origin of the most Internet users) the e-commerce stood at 0.8% of the total retail trade in 1999 and increased to 1.2% until 2002 [5]. Besides, according to Taylor Nelson Sofres (TNS) report [4], the percentage of Internet users that buy on-line grew from 27% in 2000 to 33% in 2001, but in the next year the same value stood only at 32%.

On one hand, in the United States we observe 50%-growth of trade activity in the Internet (in years 1999-2002), but on the other hand, that activity is relatively marginal.

One of the reasons of e-business slowdown in the beginning of 21st century was a slump on the Internet market. It was partly a consequence of the forecasts from the late nineties that overestimated general interest in e-commerce. The forecasts were prepared by well-known and recognised consulting corporations (e.g. Arthur Andersen or Merrill Lynch), and therefore, were treated as very probable [6].

However, one of the major barriers in the development of electronic business all over the world seems to be consumers' lack of trust [1, 4] that results from:

- lack of direct contact between consumers and vendors or bank assistants,
- fear of transaction security violation,
- fear of personal data security violation,
- a shortage of information regarding a second party and its reputation.

These issues have crucial impact on retail websites credibility perceived by consumers, and affect economical efficiency growth of the Internet suppliers.

PROBLEMS OF E-COMMERCE DEVELOPMENT IN POLAND

In Poland, business activity in the Internet seems to be much weaker than in the United States and other developed countries. For example, in 2002 the percentage of Internet user that decided to shop on-line was more than three times less than in the United States in the same year. Similar situation takes place in other East European countries [4].

We can distinguish some additional barriers that result in lower popularity of e-commerce in Poland, caused mainly by the previous political and economic system:

- poorly developed IT infrastructure and, consequently, low percentage of Internet users and limited access to net services,
- lack of appropriate legislative regulations that on one hand facilitate IT infrastructure development and, on the other hand, improve security of electronic transactions,
- low level of users' knowledge of security technologies and mechanisms applied in Internet transactions.

A final solution to these problems requires time and depends on financial abilities and legislative efficiency of the state. Therefore, this paper focuses rather on building Internet users' trust, as a first step towards solving the problems of e-commerce development in Poland, and, possibly, in other East European countries.

PREVIOUS RESEARCH PROJECTS

Factors affecting Internet users' trust and credibility of websites are a subject of many research projects carried out in recent years (e.g. Egger [1], BJ Fogg et al [2]). One of the most remarkable researches was conducted by BJ Fogg's team. They found out that a website credibility is a multidimensional function of such groups ('dimensions') of factors, as:

- real-world feeling,
- ease of use,

- expertise,
- tailoring,
- amateurism,
- trustworthiness,
- commercial implications.

Noteworthy is the explanation that 'the trustworthiness dimension of credibility captures the perceived goodness or morality of the source' and 'is defined by the terms *well-intentioned*, *truthful*, *unbiased*, and so on'. This notice is very important for describing meanings and ranges of both concepts: 'credibility' and 'trustworthiness' that in many languages (such as Polish) have almost the same or exactly the same meaning.[1]

The BJ Fogg's team conducted a research among Internet users from the USA and Finland. Both countries are treated as developed and with high level of *social trust*[2], which significantly impacts trusty behaviours in general. Therefore, there is no evidence that these factors influence
a website credibility in the same way in transition economies (where social trust level is usually rather low).

Moreover, the team was interested in factors impacting credibility of all the types of websites, not only retail ones. For those reasons, a new investigation focused on Polish users and retail websites was needed.

MODELLING CREDIBILITY OF E-SHOPS

The main assumption of the author's research was that the factors affecting a website credibility could be divided into the following groups:

- informational factors – concerning informational content of a website;
- references – from media and other customers, from real environment and the Internet, as well as certificates from trusty institutions or societies;

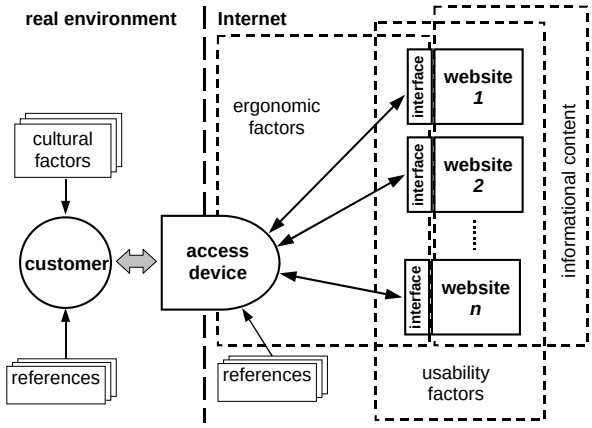


Figure 1. Main factors affecting Internet user's trust to e-shops.

- cultural factors – referring to customer's social origin, mentality, customs, inclinations and general attitude to trust;
- ergonomic factors – concerning website user interface and its compatibility with Internet access devices;
- usability factors – concerning the website and its design.

Figure 1. presents a model that illustrates how the foregoing groups of factors affect consumer's trust to e-shops and individually perceived website credibility.

One can see that consumer's trust depends on factors stemming from real environment, as well as from the Internet.

It is significant that references reach a customer from both environments. However, they can be transferred between both these separate 'worlds'. References can also be shared in a formal, as well as in an informal way (e.g. from media or directly from other customers).

On the other hand, cultural factors (in the meaning described above) occur only in reality, and they cannot be transferred directly into virtual environment. However, relevant knowledge about cultural factors can be used for designing a website interface by matching the language, appearance and way of interaction with a customer's expectations and background.

In the author's view, a suitable combination of cultural factors and informational content should partly compensate the negative influence of the low trust level. For this it will be one of the issues to be validated in the author's current research.

THE RESEARCH SCHEDULE AND ITS FIRST RESULTS

The research outcomes should answer two crucial questions:

- Which particular factors affect e-shop credibility and how do they do so?
- How much is the e-shop credibility important for a customer's decision process?

The separate investigations are planned for each question. Recently the first part of the research has been run. This

[1] In the author's opinion, e-shop credibility perceived subjectively by a customer is the reflection of his/her trust to the e-shop.

[2] Simplifying, in sociology, *social trust* means general tendency to trust other people.

part proceeds in five stages. While this article is being written the third stage finishes. Thus, some results can be published.

The 1st Stage: Identifying Credibility Factors

In the first stage, three groups of students (of 14, 12 and 22 persons) were asked to surf across three different retail web services and choose the most credible one. To obtain maximum varied and numerous set of factors each group had a different task and different set of websites to review.

Then the students had to point out the factors that had influenced a website credibility (in both ways: negatively and positively). They also had to assess intensity of each factor (in a 3-level scale).

The students identified 361 factors that were in turn grouped into 12 categories. The most frequently pointed factors belong to the following criteria: 'product description', 'ergonomics and usability', 'information about supplier' and 'web service functionality'.

Surprisingly, the least frequently factors mentioned by respondents concern transaction and data security (!).

None of the respondents pointed out cultural factors; however, in the author's opinion these factors could not be omitted (the results of the next stage partly proved that) and were taken into consideration in further stages.

On the other hand, the respondents pointed out the factors that could not be numbered among five earlier assumed groups, but exemplify very important quality issues such as functionality and reliability.

Main group of factors	Category	Number of factors / number of indications
Informational factors	information about supplier	40/147
	product description	67/222
	general information	29/67
	marketing issues	16/59
	placing orders	24/87
	information about delivery	25/67
	data and transaction security	6/9
References	references	15/109
Ergonomic and usability factors	ergonomics and usability	57/172
Quality issues	technical quality (reliability)	25/85
	web service functionality	33/127
	overall impression	24/74

Table 1. Credibility factors structure.

In most cases the students mentioned ergonomic and usability factors together; therefore these two groups have been joined into one group of factors.

Table 1. presents a structure of credibility factors divided into groups and categories. The last column in the table contents number of factors and number of their indications in each category.

The students described pointed out factors using the informal language. Frequently different descriptions were used for the same item. The students often gave the example of the same factor both in positive and negative light (e.g. 'complete product description' as a factor positively influencing credibility, and 'incomplete product description' as a negative one). For those reasons a number of factors could be reduced more than three times.

The 2nd Stage: Designing and Validation of Paper Pilot Survey

Relying on the results from the 1st stage a paper pilot survey was designed. The survey included 114 statements corresponding with the earlier identified factors. There were also included five additional statements concerning cultural factors. The survey included also two open questions concerning factors, disregarded by the author, in respondent's opinion, and general opinion about the survey.

25 students participated in that stage. The students had to make a standpoint about each statement using a 7-point Likert-type scale (from '+3', representing a strongly strengthened credibility expression, through neutral point to
'-3' representing a strongly weakened credibility expression).

The outcomes showed that the number of statements was too large. It was easy to predict, however, that allowed indicating the statements that were ambiguous, incomprehensible or redundant. It was signalized by students and also resulted from data analysis, e.g. high dispersion of answers and/or answer's character (positive/negative) which doesn't correspond with the author's intention.

The 3rd Stage: Designing and Validation of On-line Pilot Survey

The experience gained during the paper pilot survey enabled to design a new on-line one.

The number of statements was reduced to 58 and incomprehensible statements were redone. There was used the same 7-point Likert-type scale as in the paper survey.

The survey was also expanded with second part including questions of demographic data, which might correlate with respondents' opinions. In this case, the survey was completely anonymous, and therefore collecting demographic data was indispensable.

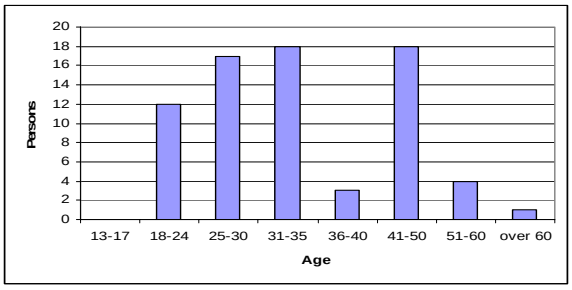


Figure 2. Respondents' age structure.

The respondents were recruited by using a 'snowball' method: a link to the survey's website was sent to familiar persons who were asked to forward the link to other persons, etc. In a one-month period (from June 20th to August 16th 2005) the site was visited 158 times, but only 73 questionnaires were completed.

In this sample of respondents:

- average age of respondents is between 31 and 35 (the Figure 2. shows the age structure of the sample),
- average education level is university with technical profile,
- average income declared by the respondents is 2-3 thousand zlotys per month (that is the average income in Poland),
- average total amount of Internet transactions is 5 in on-line auctions, and 4 in e-shops,
- 33 respondents use e-banking in full-service range and 31 only for paying fees,
- most of respondents declare moderate trust to Internet transactions.

In respondents' opinion 'information topicality' and 'data security' have the most positive impact on e-shop credibility. Important are such factors as: 'good and quick contact with e-shop personnel', 'varied paying methods'.

Ergonomic and usability factors are also perceived as highly significant. This group of factors became the most numerous after statements' number reduction. The highest assessed factors are 'website clearability and readability' and 'ease of finding information within a website'.

Negatively perceived are such factors as: 'personnel incompetence', 'lack of product price information' and 'false links on an e-shops website'.

For respondents unimportant seem 'website design' and 'date of establishing e-shop'.

The group of respondents is too small to find clear correlations between data obtained from the first and the second parts of the survey, thus the presented outcomes could be treated only as estimated. However, the main goal of this stage was to validate the survey and to check whether the survey data collecting system used in the website worked properly. In the author's opinion the results, which have been obtained so far, let suppose that the goal has been achieved entirely.

The Stages 4th and 5th: Launching the On-Line Survey and Data Analysis.

The next steps after pilot survey closure will be launching a new on-line survey and statistical analysis of the collected data.

Participants' recruitment is planned to be completed in a more effective way than for the pilot survey: a link to the survey's website will be placed on sites of two of the largest Internet portals in Poland.

CONCLUSION

The results obtained so far allow noticing that there are many similarities between credibility of a website and the quality of its user interface. The same factors – usability, functionality and reliability – influence the quality as well as the credibility.

Despite the differences between both concepts (e.g. informational content and references), one can suggest that methods used to measure and improve quality of websites may be useful for their credibility as well.

The outcomes of current research should give more details about factors – including the quality items – that crucially impact on e-shops credibility perceived by Internet users, particularly with reference to the Polish society.

THE REFERENCES

1. Egger, F.N. "Trust Me, I'm an Online Vendor": Towards a Model of Trust for E-Commerce System Design. In: G. Szwillus & T. Turner (Eds.), CHI2000 Extended Abstracts: Conference on Human Factors in Computing Systems, The Hague (The Netherlands), April 1-6, 2000, p. 101-102.
2. Fogg, B.J., Marshall, J., Laraki, O., Osipovich, A., Varma, C., Fang, N., Paul, J., Rangnekar, A., Shon, J., Swani, P., & Treinen, M. What makes Web sites credible? A report on a large quantitative study. Proceedings of CHI'01, Human Factors in Computing Systems, 2001, pp. 61-68.
3. InterCommerce Corporation. Internet survey: http://www.survey.net, 1999-2002 (visited: Oct. 05, 2003).
4. Taylor Nelson Sofres (TNS) Interactive. Global e-Commerce Report 2002. http://www.tns-global.com (visited: Aug. 25, 2005).
5. United States Department of Commerce. Retail E-Commerce Sails in Second Quarter 2002, Census Bureau Reports, Washington 2002.
6. Waszczyk M. Trust and on-line retailing. In: J. Kubka (Ed.): Economics and Values. Gdansk University of Technology 2002, pp. 131-138.