

Stellingen behorende bij het proefschrift:

What's the problem?

Studies on identifying usability problems in user tests

- 1 Iedere definitie van bruikbaarheidsproblemen waarin de 'analist' niet expliciet wordt genoemd, is incompleet (dit proefschrift).
- 2 Het bepalen van overeenkomsten en verschillen tussen door analisten gerapporteerde bruikbaarheidsproblemen is geen triviale onderneming (dit proefschrift).
- 3 In een bruikbaarheidstest de onderzoeker een 'paper prototype' laten bedienen in opdracht van de gebruiker, heeft toegevoegde waarde voor het inzicht krijgen in de denkprocessen van deze gebruiker (dit proefschrift).
- 4 Onderzoek naar de kwaliteiten van bruikbaarheidsevaluatiemethoden zou niet wezenlijk moeten verschillen van onderzoek naar de kwaliteiten van producten die met die methoden onderzocht worden.
- 5 Het voorstel van Bevan (2008) om het begrip *user experience* te definiëren door het in de ISO 13407 norm onderdeel te maken van het begrip *usability*, onderschat het belang dat *user experience* heeft voor consumentenproducten (Bevan N., 2008, *UX, Usability and ISO Standards*. CHI'08).
- 6 Het principe van de DEVAN checklist is ook bruikbaar voor het opsporen van problemen rond gebruiksbelevingen zoals plezier (Barendregt, W., 2006, *Evaluating fun and usability in computer games with children*).
- 7 Parttime promoveren is bevorderlijk voor de diepgang van promotieprojecten.
- 8 Promovendi zijn gemakkelijker te begeleiden wanneer ze zentrainingen hebben gevolgd waarin gewerkt wordt met Hisamatsu's fundamentele koan "Als nu, in deze situatie, wát je ook doet, niets zal uithalen, wat ga je dán doen?".
- 9 Het leren schrijven van haiku in de traditie van de Japanse dichter Matsuo Bashō stimuleert waardevrij observeren in de wetenschap.
- 10 Het veel gebruikte cliché "het is natuurlijk een cliché, maar..." maakt duidelijk dat het gebruiken van clichés zowel gewenst als ongewenst is.

Arnold Vermeeren, Delft, 30 Maart 2009

Deze stellingen worden opponeerbaar en verdedigbaar geacht en zijn als zodanig goedgekeurd door de promotor Prof. dr. H. de Ridder.

Propositions accompanying the thesis:

What's the problem?

Studies on identifying usability problems in user tests

- 1 Any definition of usability problems without explicitly mentioning the 'analyst' is incomplete (this thesis).
- 2 Determining differences and similarities between usability problems reported by analysts is not a trivial endeavour (this thesis).
- 3 Using a 'paper prototype' controlled by the researcher in commission of a user during a usability test, yields added value for gaining insight into the user's thought processes (this thesis).
- 4 Research into the qualities of usability evaluation methods should not be fundamentally different from research into the qualities of products that are evaluated using such methods.
- 5 Bevan's (2008) proposal to define the term *user experience* by making it part of the concept of *usability* in ISO standard 13407, underestimates the importance of *user experience* to consumer products (Bevan, N., 2008, *UX, Usability and ISO Standards, CHI'08*).
- 6 The principle of DEVAN's checklist also holds for detecting problems concerning user experiences such as fun (Barendregt, W., 2006, *Evaluating fun and usability in computer games with children*).
- 7 Pursuing a Ph.D. in part-time is beneficial for the depth of the PhD project.
- 8 PhD candidates are easier to coach after they have undergone zen trainings in which they practiced to use Hisamatsu's fundamental koan "Right here and now, if nothing you do will do, then what will you do?".
- 9 Learning to write haiku in the tradition of the Japanese poet Matsuo Bashō stimulates value-free observation in science.
- 10 The frequently used cliché "of course it is cliché, but..." indicates that using clichés is both desirable and undesirable.

Arnold Vermeeren, Delft, 30 March 2009

These propositions are considered opposable and defensible and as such have been approved by the supervisor, Prof. dr. H. de Ridder.

What's the problem?

Studies on identifying usability problems in user tests

Arnold Vermeeren

Cover design, layout and chapter overview figures: Theo Rooden (TRGW)
Printed by Ipskamp drukkers, Enschede

ISBN: 978-90-5155-051-1

© Arnold Vermeeren, 2009

Back cover: Translation of the poem “Oneindig veel problemen” © 2007, Rutger Kopland, *Verzamelde gedichten*, 2e druk 2007, p. 389. Translation by James Brockway, netherlands.poetryinternationalweb.org; edited by Willem Groenewegen, 2008. Original poem and translation used with permission.

All rights reserved. No part of this book may be reproduced or transmitted in any form or by any means electronic or mechanical, including photocopying, recording or by any information storage and retrieval system without permission from the author.

What's the problem?

Studies on identifying usability problems in user tests

Proefschrift

ter verkrijging van de graad van doctor
aan de Technische Universiteit Delft,
op gezag van de Rector Magnificus prof. dr. ir. J.T. Fokkema,
voorzitter van het College voor Promoties,
in het openbaar te verdedigen op maandag 30 maart 2009 om 12.30 uur door

Arnoldus Petrus Otto Sabina VERMEEREN

ingenieur Industrieel Ontwerpen
geboren te Oss

Dit proefschrift is goedgekeurd door de

promotor

Prof. dr. H. de Ridder

copromotor

Dr. A.J. van Doorn

Samenstelling promotiecommissie:

Rector Magnificus, voorzitter

Prof. dr. H. de Ridder, Technische Universiteit Delft, promotor

Dr. A.J. van Doorn, Technische Universiteit Delft, copromotor

Prof. dr. G. Cockton, University of Sunderland, United Kingdom

Prof. dr. K. Väänänen-Vainio-Mattila, Tampere University of Technology, Finland

Prof. dr. M.A. Neerincx, Technische Universiteit Delft

Dr. K. Hornbæk, University of Copenhagen, Denmark

Dr. J. Aasman, Franz Inc., United States of America

Prof. ir. D.J. van Eijk, Technische Universiteit Delft, reservelid

“Bashô zei: ‘Leer over de pijnboom van de pijnboom, leer over de bamboe van de bamboe’. ‘Leer’ betekent hier: dring door tot het wezen van de dingen.”

Hattori Dohô over zijn haikuleraar Bashô in ‘Sanzôshi: Akasôshi’¹

“Ik werd geïnspireerd door het verhaal van de beroemde zwaardsmiden Kan Chiang en Mo Yeh die hun volledige toewijding gaven aan het vervolmaken van hun kunst.”

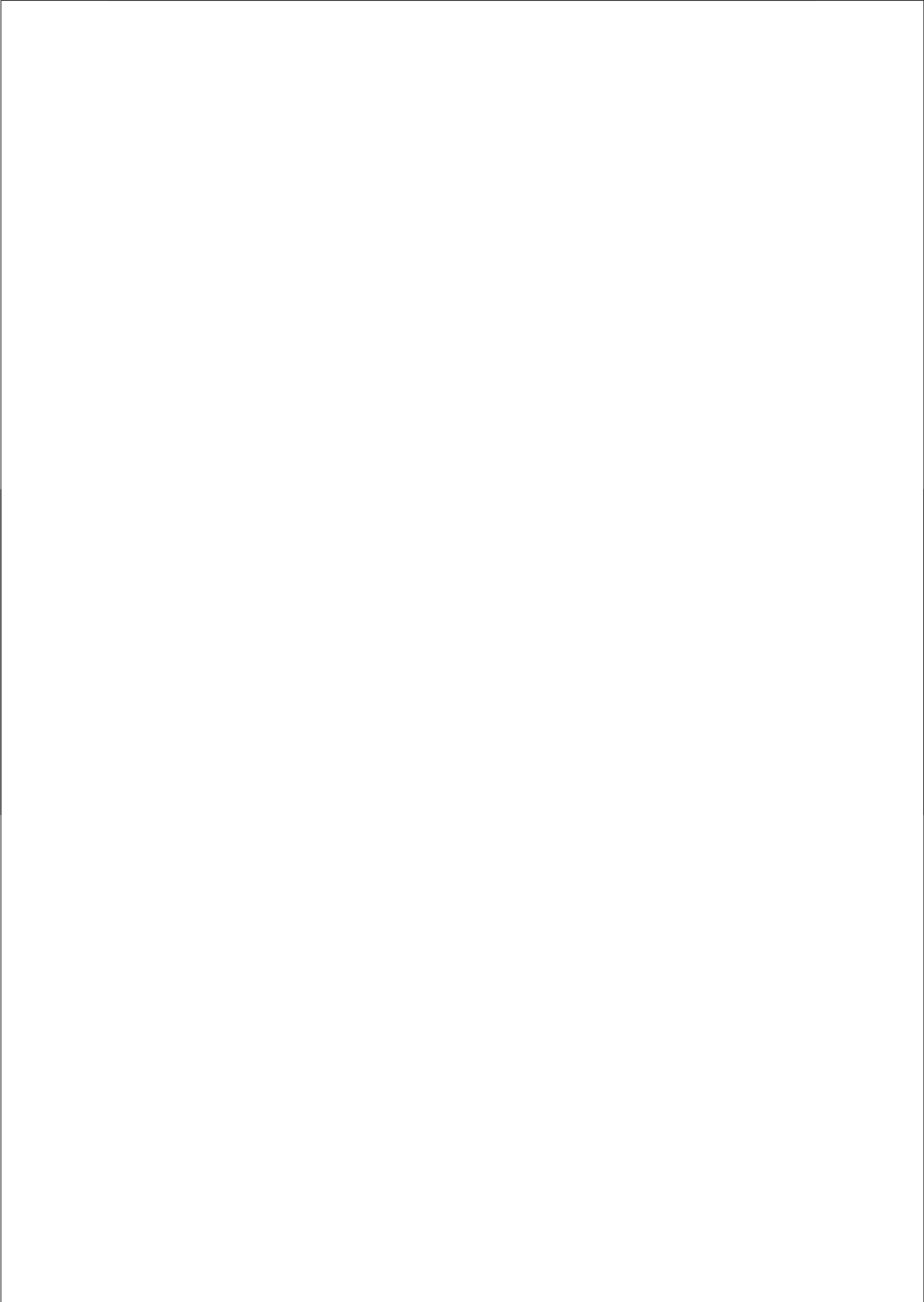
Matsuo Bashô, Het smalle pad naar het verre noorden

¹ Uit: W. Vande Walle. Basho, dichter zonder dak. Haiku en poëtische reisverhalen. Leuven: Peeters, 1985.

Table of contents

chapter 1 Introduction	9
Brief history on user-centered design	11
Usability testing	13
The evaluator effect	18
Organization of the thesis	22
Terminology used in the (Slim)DEVAN procedure	24
chapter 2 DEVAN: a tool for detailed analysis of user test data	29
Introduction	31
Analysis of the problem	32
Description of the tool	34
Evaluating the tool	45
General discussion	61
chapter 3 Usability reports for comparative studies: consistency and inspectability	65
Introduction	67
Case study	71
The reference analyses	74
Comparing the reports	79
Discussion and conclusion	97
chapter 4 Identifying problems in product tests: effects of using cardboard and software prototypes	103
Introduction	105
Methods	106
Results	117
Conclusion	127
Practitioner's take-away	128
chapter 5 Discussion and conclusion	131
Developing the (Slim)DEVAN data analysis procedure	133
(Slim)DEVAN, inspectability and consistency	133
Applying DEVAN to a comparative study on prototypes	142
Conclusion	142

References	143
Appendix A, Examples from the lab-specific analyses	151
Appendix B, Examples from the SlimDEVAN analyses	155
Summary	161
Samenvatting	165
Dankwoord / Acknowledgements	171
Curriculum Vitae	175
List of publications	177



chapter 1 Introduction

“What's the problem? Why does it not do what I want it to do?”

This is the kind of muttering researchers in a usability lab often hear when users are having trouble performing a given task with the product or prototype being tested. The activity the researcher is performing in this situation is called conducting a usability test: a test to find out what difficulties people run into when they are trying to use a (newly designed) product. During the usability test and in interviews or analyses afterwards, the researcher will try to figure out why the user experienced a given difficulty and what product characteristics contributed to that. Once the problem is understood, a designer can attempt to redesign the product's characteristics so that the chance of future users running into the same difficulty is minimized.

This approach is part of a product or software design approach that is often referred to as user-centered design. Usability tests are seen as a key technique in user-centered design approaches. Although in practice usability tests are often complemented with other methods of getting input from prospective users, or dealing with a product's context of use (e.g., participatory design, co-design, applied ethnography, contextual design; Steen, 2008), this thesis will focus on usability tests only; it deals with the methodological issues of identifying usability problems in usability tests.

What's the problem?

Studies on identifying usability problems in user tests

1 Brief history on user centered design

1.1 User-centered design principles

Before the 1980s the development of computers and software was largely an activity that was typically conducted in large companies and in laboratories of universities. Personal computers did not yet exist, professionals managed departmental mainframe or mini computers and software was mainly developed for specific users (e.g., researchers) who had some specific need for it and who were in the direct environment of the ones doing the programming. Programmers often knew the people they programmed for and knew about their tasks. Software users took for granted they had to spend some hours learning how to use the software, because the added value of having the computer functionality available was worth it.

Then in the early 1980s things gradually started to change. Human-computer interaction started to emerge as a new discipline. This was signaled for example by the organization of the first CHI conference on Human Factors of Computing Systems in 1982 as well as by a number of highly influential articles and papers for example from researchers at XEROX and IBM.

At XEROX researchers started realizing that more and more software would be built to be used by people with a different background than the programmers and that a new design approach would be needed. This was nicely phrased in a famous article about the development and testing of the Xerox Star Work Station (Bewley et al. 1983). The authors argued that a novel design approach for the Xerox Star User Interface was needed because it was one of the very first computers that had to be designed for

*“casual users who demand extensive functionality at a small training cost” [and]
“since the background of the targeted users was very different from that of the Star’s*

What's the problem?

Studies on identifying usability problems in user tests

designers, the designers' intuitions could not always be used as the criteria for an acceptable system" (Bewley et al. 1983, p72).

The field needed to become more scientific. One of the ways in which the Xerox design team dealt with that was by using a number of principles from cognitive psychology as their starting point (e.g., an explicit user's model of the system, seeing and pointing instead of remembering and typing, 'what you see is what you get'). However, the researchers at XEROX also felt that next to knowledge and models on general human information processing characteristics tools were needed in the form of 'human factors testing' of design proposals (Bewley et al. 1983). This was very much in line with another influential project that took place in the same period at IBM. Gould and Lewis (researchers at IBM) advocated a new user-centered approach at the 1983 CHI conference. In their paper they presented the following 'key principles on designing for usability' (Gould and Lewis, 1983):

- 1 *designers must understand who the users will be; this understanding is arrived at in part by directly studying their cognitive, behavioral, anthropometric, and attitudinal characteristics, and in part by studying the nature of the expected work to be accomplished.*
- 2 *a panel of expected users (e.g., secretaries) should work closely with the design team during the early formulation stages,*
- 3 *early in the development process intended users should actually use simulations and prototypes to carry out real work, and their performance and reactions should be measured,*
- 4 *when problems are found in user testing, as they will be, they must be fixed. This means design must be iterative: there must be a cycle of design, test and measure, and redesign, repeated as often as necessary.*

In the years to follow, variants of these principles appeared in journals and handbooks on human-computer interaction (e.g., Gould and Lewis, 1985; Gould, 1988).

In 1999 the ISO standard 'human-centered design processes for interactive systems' (ISO 13407, 1999) was developed (this standard is still in use). According to that standard human-centered design processes are characterized by *a) the active involvement of users and a clear understanding of user and task requirements, b) an appropriate allocation of functions between users and technology, c) the iteration of design solutions, and d) multi-disciplinary design.* According to that same standard the four human-centered design activities that should take place during a system development process are *a) to understand and specify the context of use, b) to specify the user and organizational requirements, c) to produce design solutions, and d) to evaluate designs against requirements.* With respect to activity c, the standard adds: *Users can be involved very early in the design through the use of static, paper-based mockups. This could involve presenting users with sketches of screen images of what a product/system is to look like, and asking them to try them out in a realistic context.* With respect to activity d, the standard adds: *Early in design the emphasis is on obtaining feedback that can be used to guide design, while later when a more complete prototype is available it is possible to measure whether user and organizational requirements have been met.*

Note that both in Gould and Lewis' (1983) principles as well as in the ISO 13407 standard, it is considered important to test a product with users in the form of usability testing.

2 Usability testing

Usability testing centers around studying issues of usability by having users try out a product or software. ISO 9241 part 11 (1998) defines usability as: *The extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use.* On the one hand this definition states what measures should be used for usability (i.e., effectiveness, efficiency and satisfaction); on the other hand it specifies which factors influence usability (i.e., users, goals, context of use).

A distinction can be made between *summative* and *formative* usability testing. In *summative* usability testing the usability qualities of some (product or computer) software are assessed relative to specified requirements. This is the kind of usability testing that ISO 9241-11 seems to refer to. It assumes quantitative data gathering. *Formative* usability testing aims at providing designers with input on how software can be improved from a usability perspective. In trying to do so, analysts usually search for usability problems in interactions as well as for the software properties that have caused the problems; together with other stakeholders they then prioritize their findings and fix (at least) the most important problems. This implies qualitative data analysis (e.g., inferring causes of problems based on observations), sometimes in combination with quantitative data analysis (e.g., problem counts).

In the remainder of this thesis, the focus is on formative usability tests based on identifying usability problems. Lavery et al. (1997)'s definition of usability problems (which is based on an extensive study of the literature on usability problems) is used as a starting point:

“A usability problem is an aspect of the system and/or a demand on the user, which makes it unpleasant, inefficient, onerous or impossible for the user to achieve their goals in typical usage situations (context)”.

In a later paper Lavery and Cockton (1997) added *perturbing* to the list of possible difficulties in system usage. Lavery et al. (1997) distinguished four components for any usability problem: a *cause*, a possible *breakdown* in the user's interaction, and an *outcome*, all of which happen in a *context*. In other words: within a specific context (e.g., user context, interaction context, task context), some cause (e.g., a design fault) may lead to a breakdown in the interaction (e.g., the user selecting an inappropriate action). This in turn may result in some undesired outcome (in terms of behavior and/or performance; e.g., the user's task fails, the quality of the work suffers, the user becomes irritated).

Lavery and Cockton (1997) also made a distinction between dialogue failures and knowledge mismatches. Dialogue failures equal breakdowns as defined above. Knowl-

What's the problem?

Studies on identifying usability problems in user tests

edge mismatches are defined as occurring when the user does not have the correct understanding of a particular aspect of the system or task (e.g., misinterpreting an icon on the screen). Lavery et al. (1997) use the term *breakdowns* for both, whereas Lavery and Cockton (1997) use the word *difficulty* for both. In this thesis both terms are used to include both dialogue failures and knowledge mismatches.

2.1 Usability testing in context

2.1.1 Usability testing in relation to the software development process

As has become clear in the section on user-centered design principles, usability testing is one of the essential activities in a user-centered development process. The 'Usability Engineering Lifecycle' approach (Mayhew, 1999) is a well-known example of implementing the user-centered design approach showing how the various elements of such an approach link to software engineering activities. It refers to a general development approach (rapid prototyping) as well as to the Object-Oriented Software Engineering approach OOSE (Jacobson, 1992). OOSE is based on the concept of *use cases* and distinguishes between five types of models that are created in subsequent stages: a *requirements model*, an *analysis model*, a *design model*, an *implementation model* and a *testing model*. The *requirements model* defines a system's boundaries and its functionality. The *analysis model* structures the system by defining various kinds of objects playing a role in the behavior that is modeled in the use cases. According to Mayhew's lifecycle usability, at the stage of the *analysis model* usability testing can be done using mockups, with the aim of eliminating major flaws in the design (formative testing). In Jacobson's *design model* stage the analysis model is refined and adapted to the implementation environment. Interfaces of objects and semantics of operations are defined using for example interaction diagrams and state transition diagrams. *Implementation models* consist of actual source code of the objects modeled in the design model. According to Mayhew's (1999) usability engineering lifecycle, in the stages of the *design* and *implementation model* usability testing is continued "to expand evaluation to previously unassessed subsets of functionality and categories of users, and also to refine the user interface and validate it against usability goals". Hence, in these stages tests gradually move from a formative to a more summative kind of testing. Finally, the *test model* refers to the way of testing the *implementation model* (the source code) against the requirements set in the *requirements model*. Obviously, in terms of Mayhew's lifecycle model this refers to summative usability testing for checking a design against usability requirements.

2.1.2 Usability testing in practice

Next to usability testing, other usability evaluation methods exist. Thus, in real-life development situations, developers have to decide what methods to use for evaluating software usability. Starting from the 1990s publications appeared in which usability inspection methods like Cognitive Walkthrough (Polson et al., 1992) and Heuristic Evaluation (Nielsen and Molich, 1990) were compared to usability testing. Cognitive Walk-

through is a usability inspection method that “involves hand simulation of the cognitive activities of a user, to ensure that the user can easily learn to perform tasks that the system is intended to support” (Polson et al., 1992). Heuristic evaluation involves “having a small set of evaluators examine the interface and judge its compliance with recognized usability principles (the “heuristics”)” (Nielsen, 2008).

Initially, such comparison studies mainly focused on comparing the number of usability problems found with inspection methods to those found in usability tests. One of the first influential publications making such comparisons was that of Jeffries et al. (1991). In the discussion section of that publication they concluded that their heuristic evaluation technique was best and that usability testing was second and more expensive to conduct. After this and other publications on comparing methods, people tended to interpret such conclusions as a rationale for not having to do usability testing and doing heuristic evaluations instead (Jeffries and Desurvire, 1992). Then, in 1998 an influential publication by Gray and Salzman (1998) appeared; it addressed many alleged methodological problems in a number of comparison studies. Around that time, more and more publications appeared discussing the methodological difficulties of identifying problems and comparing methods. Recently, the focus in comparing methods has shifted towards a strategy of scoping methods (i.e., determining relative strengths and weaknesses of methods; e.g., Blandford et al., 2008). Thus, instead of determining which method is best the focus shifted to the question of ‘which methods can best be combined and how?’. At the same time, attention also shifted from focusing only on a method’s productivity in finding problems, towards meeting various other constraints that may play a role in using a method in practice (e.g., Cockton, 2006; Furniss, 2008; Streefkerk et al., 2008). For example, in product development contexts, usability is positioned amongst (and has to compete with) various other values in relation to a product (e.g., Cockton, 2004) implying that the application of usability evaluation methods is also determined by the extent to which stakeholders are willing to invest in usability at all. In academic contexts, especially in those where academic studies are conducted to study the concept of usability or usability testing itself, constraints are usually very different from those in a product development context. Specifically, in an academic context time constraints are usually looser than in a product development context.

2.2 Components of a usability test setup

No matter in what context a usability test is conducted, decisions have to be made about the various components of a usability test set-up: users (i.e., test participants), goals (i.e., test tasks), context of use (e.g., usability laboratory). With respect to the participants, for example, decisions have to be made on how many participants need to be involved in a usability test and on what the most important user characteristics are for selecting participants to be involved in the test.

Driven by the need for more efficient testing a number of studies have been conducted in search for the optimal number of participants in a usability test. Formulas were derived for estimating the number of participants needed to uncover ‘most’ usability

What's the problem?

Studies on identifying usability problems in user tests

problems. Graphs resulting from such studies typically showed an asymptotic form where the asymptotic value stands for 'the total number of usability problems' (e.g., Virzi, 1992; Nielsen and Landauer, 1993, Lewis, 1994). These studies showed that in their cases five users sufficed to identify about 80 - 85% of 'all' usability problems they had found. Gross over-generalizations of such findings have led to statements like: "After the fifth user, you are wasting your time by observing the same findings repeatedly but not learning much new" (Nielsen, 2000). Later, various studies have appeared challenging such generalizations (e.g., Cockton, Lavery and Woolrych, 2003, Spool and Schroeder, 2001).

One of the most recent of such challenges is that of Lindgaard and Chattratchart (2007). Based on an analysis of data from one of the most well-known comparative usability evaluation studies involving nine usability labs (CUE-4, Molich and Dumas, 2008) they concluded that it would be better to shift the focus from the *participant* to the *task* component of usability testing. Their study showed "no significant correlation between the percentage of problems found or of new problems found and number of test users, but correlations of both variables and number of user tasks used by each usability team were significant". Lindgaard and Chattratchart concluded that "with careful participant recruitment, investing in wide task coverage is more fruitful than increasing the number of users".

Next to *users* and *tasks*, *context of use* is the third component that may affect usability. With the current rise of the number of mobile electronic devices there is an increase in the number of publications that report findings on how usability testing in the lab relates to studies conducted in a product's actual context of use (e.g., Nielsen et al., 2006; Öztoprak and Erbug, 2006; Vermeeren et al., 2008). Vermeeren et al. (2008) found that usability tests in a laboratory gave weak predictions of actual usage as measured by tracking in the actual context of use over a longer period of time. On the other hand the laboratory tests did provide detailed and rich insights into issues of usability that automated tracking of actual usage could not provide. Nielsen et al. (2006) reported to have identified significantly more usability problems in a field setting than in the laboratory and states that "this setting revealed problems with interaction style and cognitive load that were not identified in the laboratory". Öztoprak and Erbug (2006) found that in a field test situation more usability problems were found than in a laboratory test and that these were due to contextual factors in the setting, including specific unforeseen usage situations and high pressure on users because of limitations in available time.

2.3 Factors specific to formative usability testing

In preparing to conduct a usability test, decisions on the components described above have to be made, regardless whether a summative or a formative test is being prepared. However, especially in case of formative usability tests, two additional components have been shown to play an important role: prototypes and analysts¹.

2.3.1 Prototypes

In the ISO 9241-11 (1998) definition of usability the fourth component, next to the ones discussed above, is the *product*. In case of formative tests (usually taking place during the design process) *product* use is mimicked by the use of a *prototype*. In the literature there exist various distinctions between types of prototypes. Houde and Hill (1997) introduced three kinds of prototypes based on the type of design questions a prototype addresses: 1) *role prototypes* simulate or demonstrate the role a product plays in a user's life (e.g., storyboards); 2) *look & feel prototypes* are for answering design questions related to the "concrete sensory experiences of use: what the user looks at, feels and hears while using it" and 3) *implementation prototypes* serve for answering questions about techniques and components through which a product performs its function. One of the most cited categorization of prototypes is that of Nielsen (1989) for software. He identified two dimensions of prototypes, based on functionality: 1) *horizontal* prototypes simulating many of the software's functions, but only in a shallow way, and 2) *vertical* prototypes which represent only a limited part of the functionality but in much depth. Virzi (1989) added to this a third dimension that he named: *similarity of interaction*. This relates to Houde and Hill's *look and feel prototypes*. Other terms that Houde and Hill (1997) used for discussing prototypes are *resolution* and *fidelity*; *resolution* refers to the amount of detail in a prototype, whereas *fidelity* refers to how close a prototype is to the eventual design. Finally, prototypes can also be distinguished based on the medium or tools used to do the prototyping (e.g., paper prototypes, VisualBasic prototypes, Clickable pdf prototypes, etc.).

Sauer, Franke and Ruettinger (2008) provided an overview of studies comparing problems found in usability tests with a variety of prototypes and functioning products. In general, the differences in the number of identified problems were small. However, the studies Sauer, Franke and Ruettinger (2008) cited did not report on one or more of the following methodological issues: a) systematic procedures for uncovering the nature of the differences in problems by looking at their contents; b) measures for improving consistency and dealing with subjectivity; c) procedures for extracting and matching usability problems. In chapter 4 of this thesis, we report on a study in which findings from user tests with a cardboard mockup and a VisualBasic software prototype of a TV-video combination are compared to each other and to those from tests with a functioning product. Each of the prototypes was a combination of a 'look and feel' and an 'implementation' prototype, with the character of a vertical prototype rather than a horizontal prototype. Both prototypes in the study were high resolution prototypes. In terms of fidelity, navigation and sequences of commands they were of high fidelity, whereas their visual appearance, the details of the physical actions to be executed

¹ In the following the term analyst is used to refer to a person analyzing data from a usability test; the term evaluator is used as a more general term for a person using any usability evaluation method (e.g., heuristic evaluation and cognitive walkthrough).

What's the problem?

Studies on identifying usability problems in user tests

and the product's feedback (speed and visual appearance) were of lower fidelity. The lower fidelity largely stemmed from constraints imposed by characteristics of the used media for prototyping. In the study, findings obtained with prototypes and the functioning TV-video are compared on various levels of abstraction, using systematic and detailed procedures. Differences are not only compared quantitatively, but also analyzed qualitatively.

2.3.2 Analysts

In the last decade, practitioners and researchers have become aware that the analyst may also play an important role in what problems will be identified in a usability test. This is largely due to the appearance of Jacobsen, Hertzum and John's (1998) paper on the so-called evaluator effect. Jacobsen et al. found that multiple evaluators who independently analyzed the same videotaped test sessions reached an inter-evaluator agreement on extracted usability problems of only 42% (on average). The first and the second of a long series of Comparative Usability Evaluation (CUE) studies by Molich et al. (2004) reported even more dramatic figures, which may be attributed to the setup of the usability tests as well as the fact that participants in the compared sessions were different.

Historically speaking, the project that we report about in this thesis was initiated in 1996/1997 by conducting comparative studies about the effect of prototype characteristics on usability test findings (the studies that are reported in chapter 4). Many hours of videotapes had to be analyzed and the analysis was spread over a substantial period of time. We found that each time we resumed the analysis we had gained new insights and felt the urge to reconsider the things we had already done. We often wondered if the differences we had found were caused by not having done the analyses in exactly the same way as before or by not making the comparisons in the same way. However, tracing back what exact decisions were made in the analyses proved to be difficult and levels of granularity in analyzing the data differed all the time. It may be that such inconsistencies also lay at the basis of the evaluator effect as reported by Jacobsen, Hertzum and John (1998). Therefore, we decided to focus our studies on issues of inconsistencies in usability test data analyses and on traceability of usability test findings first. In the end, these issues became the central topic of this thesis.

3 The evaluator effect

Jacobsen, Hertzum and John (1998) found that multiple evaluators who independently analyzed the same videotaped test sessions managed to reach only limited inter-evaluator agreement on extracted usability problems and called this *the evaluator effect*. Hertzum and Jacobsen (2001) defined the evaluator effect somewhat broader as "*differences in evaluators' problem detecting and severity ratings*". According to Hertzum and Jacobsen (2001) one of the consequences of the evaluator effect is that "*it is highly questionable to use a thinking-aloud study with one evaluator as an authorita-*

tive statement about what problems an interface contains". They identified three aspects of usability evaluation methods as possible contributors to the evaluator effect: 1) vague goal analyses leading to variability in task scenarios, 2) vague evaluation procedures, and 3) vague problem criteria leading to anything being accepted as a usability problem. They found that several of the studies they reviewed *"have dealt with one of the three vaguenesses and can serve to illustrate that as long as the other vaguenesses remain, the evaluator effect is still substantial"*.⁴ In addition, they stated that *"A couple of the studies attempt to deal with all three vaguenesses and achieve some of the most consistent results"* suggesting that indeed the mentioned vaguenesses played a role in the occurrence of the evaluator effect. Hertzum and Jacobsen (2001) believe that the evaluator effect can never be eliminated completely. Indeed, the question is, whether the evaluator effect and inconsistencies in data analyses are insurmountable or not. Can they be reduced by using appropriate data analysis procedures?

3.1 Sources of inconsistencies in analyses

Possible sources of differences may be traced back to specific activities in detailed test analyses. The most likely ones are:

- 1 *Logging user-product interaction.* Here, possible sources of differences relate to questions like 'what exactly is logged and in what detail?'
- 2 *Transcribing verbal utterances and non-verbal behavior.* This yields possible sources of differences related to issues such as: 'What exactly is transcribed, as well as into what form. Is everything the user says transcribed, or only part of it? Is it transcribed literally or in a summative or interpretative way? What non-verbal behavior is transcribed, into what form and how detailed?'
- 3 *Inferring user's goals and intentions.* Lack of information about 'what goes on inside the user's head', can hinder understanding the interaction, as a user's goals and intentions have to be inferred from what the user does and says (possibly in combination with task scenarios given to the user). Think-aloud protocols or probed interviews directly following the interaction can aid at this stage, but are never complete. Thus, possible sources of differences come in through differences in knowledge, experience and empathy of analysts as they infer goals and intentions.
- 4 *Identifying usability problems.* Identifying as well as deciding on similarity or dissimilarity of usability problems and determining whether various behavioral difficulties in interactions actually are instances of the same usability problem or not, are largely a matter of perspective. Such decisions depend upon the specific research questions for which a usability test is conducted. Differences in detecting problems may also be caused by decisions concerning 'which behavioral events should be interpreted as indicators of problem occurrences'. For example, should misunderstandings and signs of frustration that do not lead to errors in task performance be treated as indications for problems? How about redundant actions without negative side effects in terms of performance?

What's the problem?

Studies on identifying usability problems in user tests

3.2 Can and should the number of differences be reduced?

This brief overview makes clear that sources of differences may be found in all stages of a usability test data analysis. Then it becomes important to determine how much these differences can be reduced and whether this is desirable. Differences such as those caused by fatigue, lack of vigilance or distraction of analysts should be reduced by using appropriate procedures and tools. Note that such differences may also occur in cases where analysts analyze the same data twice. Other differences may relate to analysts' beliefs, values and preferences, for instance when they infer users' goals and intentions or when they identify and categorize problems. By training the analysts it may be possible that also this kind of differences can be reduced (but most likely not eliminated). However, one has to realize that this may mean biasing the analysts, which is sometimes undesirable. For example, if the aim is to find as many usability problems as possible, training the analysts to all see the same kinds of problems would reduce the overall number of problems found across analysts.

3.3 Consistency in relation to comparing problems

The main conclusion from the considerations above is that the analyst will always be an integral part of the instrument for identifying usability problems. But what is the consequence of this observation for conducting studies in which usability problems are compared? In particular, what is the consequence for the consistency in relation to comparing problems?

Studies on comparing usability problems are often about comparing findings from using different Usability Evaluation Methods (UEMs) in order to determine the specific quality aspects of the UEMs. Hartson, Andre and Williges (2001) discuss four quality criteria for usability evaluation methods, namely *thoroughness*, *validity*, *effectiveness*, and *reliability*. They propose how to determine thoroughness, validity and effectiveness based on counting usability problems. In contrast, they do not specify the concept of *reliability* in terms of how to calculate it. Instead they describe *reliability* primarily in terms of evaluator agreement, namely as "... a measure of the consistency of usability testing results across different users of the UEMs (evaluators)" (p. 396). At the same time they mention *individual reliability*. From their publication we infer that this concerns consistency of UEM results in cases where an evaluator applies a UEM multiple times on the same material (i.e., within-evaluator consistency). So there are two kinds of consistency. First consistency of individual analysts is discussed, followed by a discussion on consistency across analysts.

3.3.1 Consistency of individual analysts

Guba and Lincoln (1989) state that the establishment of reliability "... typically rests on replication, assuming that every repetition of the same, or equivalent, instruments to the same phenomena will yield similar measurements" (p. 235). At the same time, they state that in research based on a naturalistic research paradigm, where by definition

measurements cannot be exactly repeated, the issue of reliability (or dependability, as it is often referred to in that context) is dealt with mainly by assuring that the used process is "... an established, trackable, and documentable process," so that outside reviewers "... can explore the process, judge the decisions that were made, and understand what salient factors in the context led the evaluator to the decisions and interpretations made" (Guba and Lincoln, 1989, p. 242).

Because in usability testing the analyst forms part of the instrument for identifying problems, reliability concepts based on 'repeated measurements' are problematic: the analyst can never be assumed to forget everything about the previous analysis, and not to gain any relevant, additional knowledge or experience affecting his/her perception of interactions. In the remainder of this thesis the term 'reliability' will be avoided. Consistency of individual analysts will be dealt with by using the repeated measures approach (within-analyst consistency) taking into account Guba and Lincoln's (1989) advice to use established, documentable and traceable processes.

3.3.2 Consistency of findings across multiple analysts

According to Guba and Lincoln (1989) in a naturalistic research paradigm one may not assume that methods can prevent the inquirer (even inadvertently) introducing *subjectivity* in findings. Assurances of integrity of findings are rooted in the data themselves. In other words, the starting point is that (at least some degree of) subjectivity is acknowledged in data analysis and should be dealt with properly. In their view this means that both the original data and the processes used to compress these data should be available for inspection and confirmation by outside reviewers of the study. For this they adopt a criterion of *confirmability*. Probably, the term *inspectability* would be more appropriate, as the data and the processes should not be available for confirmation only but also for detailed discussion.

The fact that in extracting usability problems the analyst forms part of the measurement instrument means that subjectivity will have to be dealt with when comparing problems across analysts. In the remainder of this thesis, issues of subjectivity will be approached 1) by using measures of agreement between analysts and 2) by adhering to Guba and Lincoln's (1989) advice that it should always be possible to trace back on what data the final findings are based and how the primary observations transformed into these findings (i.e., findings should be inspectable).

3.4 Data analysis procedures for comparative studies

From the previous it may be concluded that documentable and traceable data analysis procedures are needed. We developed such a procedure by combining elements of two existing frameworks, namely Sutcliffe et al.'s (2000) Model Mismatch Analysis (MMA) and Cockton and Lavery's (1999) SUPLEX (Structured Usability Problem EXtraction) framework.

Sutcliffe et al.'s (2000) MMA is an approach that deals with *detecting* usability problems. It helps detecting problems based on their surface manifestations, while causes

What's the problem?

Studies on identifying usability problems in user tests

of problems are inferred based on a walkthrough using a theoretical model of interaction. Cockton and Lavery (1999) propose the SUPEX framework for usability test data analysis. This framework covers the entire usability data analysis process from beginning to end. However, it is a framework, rather than a method or technique that can directly be implemented. The framework is based on the following data analysis steps: 1) isolation of relevant episodes, 2) analysis of relevant difficulties, 3) causal analysis and 4) recommendation generation. The procedure we developed is based on the MMA procedure for detecting problems and the first two stages of Cockton and Lavery's (1999) SUPEX framework. To meet the criteria of inspectability and traceability we extended these two data analysis procedures.

In the literature still other frameworks can be found dealing with usability problems. One of the well-known data analysis approaches is Andre et al.'s (2001) User Action Framework (UAF). UAF classifies usability problems and causes in a very extensive way, based on a large, hierarchically structured database of usability problems. The database originates from collecting and categorizing findings from a large number of usability tests. Another classification system has been proposed by Zapf et al. (1992). Their taxonomy is inspired by models of Rasmussen (1982), Hacker (1986), Norman (1986) and others. Like UAF, this taxonomy mainly deals with systematically *categorizing* usability problems, rather than with systematically *detecting* the occurrence of problems. Since we developed a procedure that stops before the stage of causal analysis and categorization, we didn't need to incorporate frameworks like UAF and Zapf et al.'s taxonomy in our procedure. However, it was our aim to develop our procedure such that its findings could be further analyzed by the latter frameworks.

4 Organization of the thesis

4.1 Main objectives of the thesis

Main objectives of the study reported in this thesis are to analyze consistency in user test data analysis and devise a way of dealing with inconsistencies, by:

- 1 developing a data analysis procedure that makes the analysis documentable, allows tracing back of usability problems to the original observations, and that provides insight into the process of transforming data from observations into usability problems.
- 2 studying to what extent the developed procedure can expose possible causes of inconsistencies and reduce less persistent differences (i.e., those caused by fatigue, lack of vigilance and distraction).
- 3 applying the procedure to find out whether it can make the data from the comparative 'prototypes' study inspectable, so that differences in identified usability problems can be traced back to characteristics of the prototyping situations.

The empirical work conducted to realize the objectives of this thesis is reported in chapters 2, 3 and 4. The studies that were used to develop and try out the new data analysis procedure were conducted in 1996/1997; the development of the procedure itself took place around 2000. Chapter 2 was published in the journal *Behaviour & Information Technology* in 2002 (Vermeeren et al. 2002). The content of the article is reprinted here, without changes other than to its visual appearance, a few corrected typing errors and a few inserted words needed to make some of the sentences more clear. The studies for chapter 3 were conducted in 2003/2004 and were published in the journal *Human-Computer Interaction* (Vermeeren et al. 2008). There were no changes to the article other than its visual appearance, a graphically revised figure 3.4, a few corrected typing errors and a few inserted words needed to make the text more clear. Finally, the studies reported in chapter 4 were conducted in 1996/1997; processing and analyzing the data was done in the years 2006/2008. In October 2008, this chapter was submitted to the *Journal of Usability Studies*.

Although the product that we used in the 1996/1997 study may be outdated, we believe that the findings from the study are still relevant. The detailed analyses aim at tracing back quantitative differences to more general characteristics of prototypes and find out how these affect user behavior and the occurrence of usability problems. This makes the results from the comparisons potentially applicable to other situations in which products and prototypes are used.

4.2 Research approach and structure of the thesis

The research approach taken in this thesis may be characterized as constructive research (Kasanen, Lukka and Siitonen, 1993). Constructive research deals with the creation of entities (e.g., models, diagrams, plans, etc.) that produce solutions to explicit problems, and their practical usefulness can be demonstrated through the actual implementation of the solution (Kasanen, Lukka and Siitonen, 1993). To be considered constructive research, the research must combine problem solving and theoretical knowledge. Kasanen, Lukka and Siitonen (1993) propose six steps for conducting constructive research: 1. *Find a practically relevant problem which also has research potential*; 2. *Obtain a general and comprehensive understanding of the topic*. 3. *Innovate, i.e., construct a solution idea*. 4. *Demonstrate that the solution works*. 5. *Show the theoretical connections and the research contribution of the solution concept*. 6. *Examine the scope of applicability of the solution*.

Constructive approaches are usually associated with case studies and qualitative methods, but quantitative methods are also used (Kasanen, Lukka and Siitonen, 1993). The preceding part of this chapter explained our steps 1 and 2. The development of a candidate solution (step 3) is described in chapter 2, along with an evaluation of 'how well it works' (step 4). The solution is called the DEVAN (DEtailed Video ANalysis) data analysis procedure. For the development of DEVAN usability tests of a programmable home thermostat were used as a case; evaluation of DEVAN was based on data analyses of two sessions from usability tests of a combined TV-video recorder. The mentioned case

What's the problem?

Studies on identifying usability problems in user tests

studies were conducted in an academic context. The evaluations made clear that for use in a product development context, DEVAN would be too time-consuming to use. Chapter 3 is based on a case-study with commercial usability labs working in an academic context. Possibilities for using and evaluating DEVAN were more constrained than in a purely academic context. Thus, a simplified version of DEVAN, named SlimDEVAN was developed and evaluated. Evaluation was done based on data from a case study in which usability tests were conducted on the digital interface of an oven. This evaluation included demonstrating how SlimDEVAN connects to theoretical notions like (in)consistency of usability test data analyses within and across analysts and provided additional insights into why such inconsistencies occurred and in how to deal with them (cf. step 5, Kasanen, Lukka and Siitonen, 1993).

Chapter 4 studies the applicability of DEVAN as a analytical instrument in answering a scientific research question (step 6). It presents a study on how prototype characteristics affect results of usability tests with (prototypes of) a TV-video combination. This study was conducted in an academic context. Because of this and the fact that in the mean time digital video and dedicated software tools for marking video data had become available it was possible to use the original DEVAN again (instead of SlimDEVAN). Next to the fact that the study reported in chapter 4 contributes to step 6 (Kasanen, Lukka and Siitonen, 1993) the study also contributes to theoretical insights into consequences of using high-level, quantitative task performance measures only, instead of combining these with qualitative analyses (step 5).

Chapter 5 summarizes the findings from chapters 2, 3, and 4 in view of the main objectives stated in this chapter. In addition, it relates the findings to other publications and presents suggestions for future work.

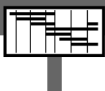






5 Terminology used in the (Slim)DEVAN procedure

"The fish trap exists because of the fish; once you've gotten the fish, you can forget the trap. The rabbit snare exists because of the rabbit; once you've gotten the rabbit, you can forget the snare. Words exist because of meaning; once you've gotten the meaning, you can forget the words. Where can I find a man who has forgotten words so I can have a word with him?"
(Zhuangzi, ~300-400 BC, in Watson, 1968)

In this thesis terms like *problems* and *difficulties* are frequently used. In this section these and other terms used in (Slim)DEVAN are defined and related to each other. Table 1 depicts the general steps in the process of conducting a usability test, analyzing its resulting data and using the data. As this thesis is about usability test data analysis, the focus will be on the analysis steps. The left-hand column depicts the steps as proposed in DEVAN, the right-hand column does so for SlimDEVAN.

After preparing and conducting a usability test (table 1, steps 1 and 2) the next step is one of making sense of the interaction. In (Slim)DEVAN this includes transcribing user actions and expressions into an interaction overview format. Table 1 summarizes the elements contained in DEVAN and SlimDEVAN interaction overviews (table 1, step 3).

Table 1 Overview of the general process of conducting usability tests, analyzing the results and using the findings for redesign or research purposes. Characteristics of how each stage is implemented in DEVAN and SlimDEVAN are summarized.

	1. Preparing usability test Not part of (Slim)DEVAN	
	2. Conducting usability test Not part of (Slim)DEVAN	
	DEVAN Elements of interaction overviews: <ul style="list-style-type: none"> Start of new (sub) task Actions with time codes Verbal and nonverbal user expressions Segmentation of interaction based on pause times Descriptions of interaction segments 	SlimDEVAN Elements of interaction overviews: <ul style="list-style-type: none"> Start of new (sub) task Actions with time codes Verbal and nonverbal user expressions
	4. Detecting difficult moments Checklist of signal event types in two categories: <ul style="list-style-type: none"> actions on products verbal and nonverbal user expressions 	Checklist of signal event types in two categories: <ul style="list-style-type: none"> actions on products verbal and nonverbal user expressions
	5. Describing difficulties Components of difficulty descriptions: <ul style="list-style-type: none"> Time code Signal event type code Free-form description of difficulty Copy of observed event from interaction overview Product mode Interaction segment 	This stage has merged with stage 6 (Describing usability problems)
	Not part of DEVAN	6. Describing usability problems Components of problem descriptions: <ul style="list-style-type: none"> Time code Signal event type Description of difficulty Likely cause of difficulty or suggest product change
	7. Using results to redesign or compare Not part of (Slim)DEVAN	

What's the problem?

Studies on identifying usability problems in user tests

In most cases, one of the major purposes of formative usability tests is to come up with a list of usability problems that can then be used to improve the tested product or that can be further processed for research purposes. It is important to disambiguate what is meant by the term *usability problem*. As mentioned before, the starting point for this is the definition that Lavery, Cockton and Atkinson (1997) arrived at after an extensive study on problem definitions. Lavery, Cockton and Atkinson (1997) distinguished between four components for any usability problem: a *cause*, a possible *breakdown* in the user's interaction, and an *outcome*, all of which happen in a *context*. In other words: within a specific *context* (e.g., user context, interaction context, task context), some *cause* (e.g., a design fault), may lead to a *breakdown* in the interaction (e.g., the user selecting an inappropriate action). This in turn may result in some undesired *outcome* (in terms of behavior and/or performance; e.g., the user's task fails, the quality of the work suffers, the user becomes irritated).

Furthermore, Lavery and Cockton (1997) make a distinction between dialogue failures and knowledge mismatches. Dialogue failures equal *breakdowns* as defined above. Knowledge mismatches are defined as occurring when the user does not have the correct understanding of a particular aspect of the system or task (e.g., misinterpreting an icon on the screen). Lavery, Cockton and Atkinson (1997) use the term *breakdowns* for both, whereas Lavery and Cockton (1997) use the word *difficulty* for both. In this thesis both terms are used to include both dialogue failures and knowledge mismatches.

After creating the interaction overview, the analysis of usability data usually consists of detecting at what moments in the interaction users have experienced difficulties. In most cases when users experience difficulties there are perceivable *signal events* that can make an analyst aware of them. The most obvious signal is when a user is asked to perform a certain task and presses the wrong button. In that case, the action the user performs on the product forms the signal that there is a problem. Such signals will be referred to as *action-based signal events* or *action signals*. Other signals may be perceived based on a user's verbal or nonverbal behavior. For example, users saying: "oops! this is wrong" or showing surprise on their face. Such signal events will be referred to as *expression based signal events*, or *expression signals*. The example above also illustrates that one single *difficulty* may be revealed by multiple *signal events*. In DEVAN as well as in SlimDEVAN a checklist of *signal event types* is used to detect difficulties in interactions. This checklist basically operationalizes what makes difficulties and outcomes of difficulties observable and is based on Sutcliffe et al.'s (2000) taxonomy of problem phenotypes.

At some points in the thesis the term '*difficult moments*' is used. This refers to the moment in an interaction at which a *difficulty* occurs. In such cases, the term is used to stress that the focus is at the *occurrence* of a *difficulty* in a specific session, and not for example at the *category* of *difficulties*. Nowhere in the thesis are *difficulties* categorized, so descriptions of *difficulties* and *difficult moments* are identical. In table 1 the stage of detecting *difficulties* in interactions is step 4.

After having detected the difficulties in an interaction, the *signal events* are described in a specified format and listed. *Signal events* referring to the same *difficult moment* are

clustered and together form the description of a *difficulty*. Descriptions of *difficulties* come close to descriptions of *usability problems* but they are not the same. A difficulty description obviously contains a description of the difficulty itself or its outcome, as well as some information about (or reference to) the context in which it occurred (task context, product mode context). The difference with usability problems is that in our definition of *usability problems* assumed *causes* of difficulties should also be included in the descriptions. The process of assigning *causes* to *difficulties* is not part of DEVAN, so the ultimate result of a DEVAN analysis is a list of *difficulties*. SlimDEVAN analyses result in lists of usability problems, as its format for describing difficulties stimulates analysts to assign *causes* to *difficulties*. However it does so without proposing a procedure for it. The stages of 'describing difficulties' and 'describing usability problems' are referred to with these exact words in table 1 (steps 5 and 6, respectively).

The framework of table 1 (the icons and stages) will be used before each chapter for summarizing the experimental work done in it. This provides the possibility to get a quick overview of what the various studies have in common as well as how they differ from each other.

What's the problem?

Studies on identifying usability problems in user tests

chapter 2 **DEVAN: a tool for detailed video analysis of user test data**

abstract

A tool was developed for structured and detailed analysis of video data from user tests of interactive systems. It makes use of a table format for representing an interaction at multiple levels of abstraction. Interactions are segmented based on threshold times for pauses between actions. Usability problems are found using a list of observable indications for the occurrence of problems.

The tool was evaluated by having two analysts apply it to three data sets from user tests on two different products. The segmentation technique proved to yield meaningful segments that helped in understanding the interaction. The interaction table was explicit enough to discuss in detail what had caused the differences in the analysts' lists of usability problems. The results suggested that the majority of differences were caused by unavoidable differences in interpretations of subjects' behavior and that only minor improvements should be expected by refining the tool.

This chapter was published as:

Vermeeren, A.P.O.S., den Bouwmeester, K., Aasman, J., & de Ridder, H. (2002). DEVAN: A tool for detailed video analysis of user test data. *Behaviour & Information Technology*, 21 (6), 403-423.

What's the problem?

Studies on identifying usability problems in user tests

Overview of characteristics of the studies reported in chapter 2

Two products
· Honeywell home thermostat
· Philips TV-video combination



1. Preparing usability test

Test prepared by **AUTHOR OF THESIS**



2. Conducting usability test

- Thermostat test (one participant session) facilitated by **AUTHOR OF THESIS**
- TV-video combination test (two participant sessions) facilitated by **ASSISTANT**

All sessions were video recorded.



3. Creating interaction overview

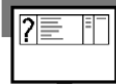
Overviews of all sessions created (using DEVAN), by **AUTHOR OF THESIS** and **ANOTHER RESEARCHER** working independently

N.B. Both the author of the thesis and the other researcher are developers of DEVAN



4. Detecting difficult moments

Detected (using DEVAN), by **AUTHOR OF THESIS** and **ANOTHER RESEARCHER** working independently



5. Describing difficulties

Described (using DEVAN), by **AUTHOR OF THESIS** and **ANOTHER RESEARCHER** working independently



6. Describing usability problems

No problem descriptions were made



7. Using results to redesign or compare

- Analysis of DEVAN's threshold pause times by **AUTHOR OF THESIS**
- Analysis of DEVAN's signal events in relation to difficulties by **AUTHOR OF THESIS**
- Comparison of described difficulties and evaluation of explicitness by **AUTHOR OF THESIS**

Making comparisons involved tracing back difficulties to single moments of difficulty.

1 Introduction

In the process of developing products it is hard to anticipate future usability problems. This is especially so for interactive systems (including consumer electronics and professional applications), where user-product interaction usually consists of interactive sequences of actions and feedback. Therefore, in such cases, user tests are usually conducted during the design process. In research on usability evaluation methods, as well as in design practice, user testing is often considered the best technique for getting insights into usability problems. For example, Nielsen (1997) states that it *“is the most fundamental usability method and is in some sense irreplaceable, since it provides direct information about how people use computers and what their exact problems are with the concrete interface being tested”*. In research, user testing is sometimes considered a *“yardstick for other evaluation methods”* (Jacobsen, Hertzum and John, 1998), or a *“gold standard against which to judge other user evaluation methods”* (Newman 1998). However, there is no standard procedure for running user tests, it is a general approach rather than a specific technique. Also, there is no standard way of analyzing data from user tests. In user testing, many decisions have to be taken of which the consequences are unclear. Moreover, such decisions are often taken implicitly. Thus, outcomes of user tests depend largely upon the knowledge and experience of the decision makers and it is not surprising that different user tests, conducted by different specialists, yield (partly) different usability problems (see for example: Jacobsen, Hertzum and John, 1998, Molich et al. 1998, Molich et al. 1999, Rowe et al. 1994, Rooden and Kanis 2000). It is no problem that decisions on the set-up of a test influence its outcomes, as long as it is clear what the influence consists of. However, for many decisions this is far from clear. To be able to improve the quality of user test outcomes in general, it is important to make explicit what the influence consists of.

What's the problem?

Studies on identifying usability problems in user tests

In a study by Vermeeren (1999), it was investigated what the influence is of providing subjects with specific types of tasks. In the study, half of the subjects were given structured tasks to perform, whilst the other half was just asked to spend 25 minutes trying out the product. The results suggested that in terms of uncovered usability problems there were hardly any differences. However, comparing results proved to be rather difficult. Most likely, this was due to the fact that no detailed and explicit data analysis technique was used to ensure that all parts of the data had received due attention. Furthermore, there had been no guarantee that the same (or at least an explicit) focus and perspective was applied to all parts of the data. To empirically study how the set-up of a user test influences its outcomes, video data from user tests have to be analyzed in a structured and detailed way and decisions taken during the analysis have to be made explicit.

In the present paper, a tool is presented for this. The tool (DEVAN: DEtailed Video ANalysis) distinguishes between two stages of analysis. In the first stage, observations are transcribed to an interaction table that represents a subject's interaction with the tested product. In the second stage, the interaction is analyzed in detail to locate events that indicate the occurrence of a usability problem.

In section two of the present article, the activity of analyzing video data is described. This results in a general specification of the functionality of the tool. Section three describes the tool in detail, focusing on pragmatic aspects of applying the tool, as well as on theoretical considerations underlying it. Section four discusses an empirical evaluation of the tool. The evaluation is based on application of the tool to three data sets from user tests on two different products. The tool was used by two analysts working independently.

2 Analysis of the problem

For analyzing user test data in a scientific context, typically hours of videotapes from many different test sessions are to be watched and understood. Usability problems have to be extracted and compared across users as well as across test conditions. Such analyses are extremely time-consuming and laborious. They may easily take weeks or months. Finding and comparing interaction episodes across users and conditions can not be done solely based on a researcher's memory or roughly indexed tapes. Repeatedly, a researcher will have to remember what it was that the subject was trying to do and how far s/he had proceeded in performing the task. Thus, it is difficult to keep an overview of which parts on which tapes have already been studied and from what perspective. In such situations, it is almost impossible to treat all data parts with due attention and from the same (or at least from an explicit) perspective or focus. Therefore, a tool like DEVAN should provide detailed, temporal overviews of videotapes: it should make the detailed contents of tapes more accessible. In addition, it should provide an overview of events indicating the occurrence of a usability problem.

2.1 Functional requirements for the data analysis tool

The aim of the tool is to structure the data analysis process, to make it explicit and to create overviews of video contents from user test sessions. The overviews should be detailed and complete, so that they assist in quickly finding and understanding relevant video fragments, or that they provide enough information so that consulting a videotape is not always necessary. Additionally, they should provide quick access to events that indicate the occurrence of a usability problem.

Software tools exist for getting quick access to videotapes (e.g., The Observer Video-Pro, Noldus 2002). Such tools generally provide technical facilities to mark events or episodes on videotapes and to perform statistical analyses. However, they do not provide the kind of multiple level interaction overviews (as suggested by for example Cockton and Lavery 1999) that are needed for methodological studies on user testing. In addition, they provide no guidance on what events may indicate the occurrence of a usability problem.

In order to develop DEVAN typical data analysis activities were identified. Below, it is indicated, how issues of subjectivity play a role in each of these activities, as well as how DEVAN should assist in coping with these issues.

Logging user-product interaction. *Issues of subjectivity include:* What exactly is logged and in what detail? *Requirements for the tool:* The tool should prescribe what actions on the product have to be logged and what else should be logged.

Transcribing verbal utterances and non-verbal behavior. *Issues of subjectivity include:* What to transcribe, as well as to what form. Should everything the user says be transcribed, or only part of it? Should it be transcribed literally or in a summative or interpretative way? What non-verbal behavior should be transcribed, to what form and in what detail? *Requirements for the tool:* The tool should guide the analyst in deciding which verbal utterances to transcribe, as well as how. In addition, the tool should inform the analyst how to treat non-verbal behavior like facial expressions of frustration, confusion, etc.

Understanding the interaction: inferring goals and intentions. By lack of information about 'what goes on inside the user's head', goals and intentions have to be inferred. Think-aloud protocols or stimulated interviews directly following the interaction can aid at this stage, but are never complete. *Issues of subjectivity come in implicitly* through knowledge, experience and empathy of analysts as they infer goals and intentions. *Requirements for the tool:* The tool should assist in creating a representation of the interaction. Through this representation, analysts should be able to make explicit how they understood specific parts of the interaction. Preferably, this should be possible at various levels of abstraction (i.e. intentions for individual actions, as well as for longer episodes).

Finding usability problems. Defining types of usability problems, as well as deciding on similarity or dissimilarity of usability problems is largely a matter of perspective. It depends upon the specific research questions for which a user test is conducted. As DEVAN is meant to be a tool for more general use, these issues will not be included in

What's the problem?

Studies on identifying usability problems in user tests

the tool. However, given a general definition of usability problems, locating events that can serve as symptoms of the occurrence of some problem might be less prone to subjectivity. Therefore, only the latter is included in the tool. *Issues of subjectivity include:* what types of events are used as indications for the occurrence of a usability problem. For example, should misunderstandings and signs of frustration that do not lead to errors in task performance be treated as indications for problems? How about redundant actions without negative side effects in terms of performance? *Requirements for the tool:* Determining event types that serve as indications for usability problems should be based on a general definition of what constitutes a 'usability problem'. This definition should discuss behavioral and performance (i.e. observable) consequences of usability problems. The tool should assist in recognizing such consequences in the interaction and it should provide for a consistent way of describing them.

3 Description of the tool

The tool (DEVAN) will be described in three subsequent sections. Theoretical issues will be described, along with the more pragmatic issues. Each section highlights a relevant aspect of the tool. These aspects are:

- 1 What should an interaction overview look like? How should the interaction be represented?
- 2 How can usability problems be recognized? What types of events can be used as indications for the occurrence of a usability problem?
- 3 What procedure is followed in the analysis? What is the order of activities to be performed?

3.1 What should an interaction overview look like?

DEVAN should provide detailed overviews of video data at multiple levels of abstraction and link them to the detailed contents of the tapes. The overviews should enable analysts to make their interpretations and perspectives explicit on all levels of abstraction.

In the following, the format for interaction overviews (see figure 1) is discussed, starting with its theoretical underpinnings.

3.1.1 Theoretical considerations

The format for representing the interaction is based on the format used in Cockton and Lavery's (1999) Structured Usability Problem EXtraction (SUPEX) framework. SUPEX' interaction representation format consists of a multiple column table of which the leftmost column is used for logging the interaction. The logging column is segmented into basic episodes. The other columns of the SUPEX table overlay these basic episodes with more abstracted representations of the interaction. With respect to the basic epi-

sodes, Cockton and Lavery (1999) found that “in a generic research context, new basic episodes begin when the system is in a stable state after the user pauses or changes focus”. This is based on concepts taken from activity theory (Nardi 1996) where pauses and users’ utterances expressing goals are the main observable evidence of user consciousness. It means that conscious user intent or concentration are taken as criterion for segmenting the interaction into basic episodes. Unfortunately, Cockton and Lavery (1999) provide no suggestions for specific threshold times that are useful for segmentation. To derive threshold pause times for use in DEVAN, inspiration was sought in the field of cognitive science, more specifically in the Keystroke Level Model (Card, Moran and Newell 1990) and the Unified Theories of Cognition (Newell 1990).

time stamp	log	interaction segments	context	breakdown indications
0:00:00	action	description of interaction segment	task 1 description	code
0:00:12	action			
0:00:13	action			
0:00:19	action	description of interaction segment	5	code
0:00:21	action			
0:00:22	action			
0:00:23	action	4	7	code
0:00:24	action			
0:00:25	action			
0:00:44	action	description of interaction segment	task 2 description	6
0:00:46	action			
0:00:49	action			
0:00:50	action	(verbal utterance)		code

1 column for logging user-product interaction 2 primary boundary, indicating the start of a new interaction segment 3 secondary boundary, indicating the possible start of a new interaction segment 4 column for interaction segment boundaries and descriptions 5 column for task descriptions and descriptions of intermediate level episodes 6 column for breakdown indication type codes 7 event marked as breakdown indication

Figure 1 General format for the interaction overview table.

The Keystroke Level Model (KLM) makes use of the concept of unit tasks. These are defined as follows: “Given a large task, (...), a user will break it into a series of small, cognitively manageable, quasi-independent tasks, which we call unit tasks”. In addition, KLM uses the concept of methods. A method is defined as “... a sequence of system commands for executing a unit task that forms a well-integrated (‘compiled’) segment of a user’s behavior”. Furthermore, KLM uses the concept of chunks. With respect to chunks

What's the problem?

Studies on identifying usability problems in user tests

Card, Moran and Newell (1990) state that *“the user cognitively organizes his methods according to chunks, which usually reflect syntactic constituents of the system's command language. Hence, the user mentally prepares for the next chunk, not just the next operation. It follows that in executing methods the user is more likely to pause between chunks than within chunks”*. This statement suggests that, at the lowest cognitive level, the length of pause times is related to starting new chunks. In addition, it suggests that chunks reflect syntactic constituents of a product's interaction language. Thus, in searching for useful threshold pause times, one should take into account the extent to which the syntax of a product's interaction language is reflected in the resulting interaction segments.

The Keystroke Level Model (KLM) provides estimates for the time expert users need to perform routine tasks. It assumes that a *unit task* consists of two subsequent parts: 1) *acquisition of the task* and 2) *execution of the acquired task*. *“During acquisition, the user builds a mental representation of the task, and during execution, the user calls on the system facilities to accomplish the task. The total time to do a unit task is the sum of the time for these parts”* (Card, Moran and Newell 1990). Given a *method* for executing a unit task, the Keystroke Level Model predicts the time needed for it. Task acquisition times would seem most appropriate for determining threshold pause times. However, Card, Moran and Newell state that: *“Task acquisition times are highly variable, except in special situations (such as a manuscript interpretation task) and we can say little yet about predicting them”*. Therefore, for determining threshold times in DEVAN, an alternative procedure is used. The starting point of the procedure is that, ideally, each chunk should become a segment in the interaction. Thus, determining threshold pause times equals determining when a chunk starts. Therefore, to make time estimates for threshold pause times means to estimate what time is minimally needed for the execution of a chunk that consists of one single action. If a measured pause time is shorter than the threshold pause time, it is unlikely that at that point a new chunk starts (at least according to KLM), but if the measured time is longer, one does not know. Using Card et al.'s parameters, task execution times for a chunk vary between 1.83 seconds (for users who's speed is comparable to that of a typist typing 135 words per minute) and 2.95 seconds (for users who's speed is comparable to that of a typist who is unfamiliar with a regular computer keyboard). Thus, based on task execution times only, threshold pause times in the range of 1.8 to 3.0 might yield useful segmentations. See table 1 for details of task execution time calculations.

Another useful strategy based on the KLM could be to include a specific (reasonably predictable) part of task acquisition in the calculations (in addition to the task execution part). For example, it is reasonably predictable that an expert typist interpreting a manuscript needs (according to the KLM) an average of at least 1.8 seconds just for reading something from the manuscript. This might be comparable to a situation in which a subject in a user test has to look up something in a given task. If, in addition, the assumed expert has to scan the CRT-display for locating the position from which to continue entering text, the time needed becomes 4.0 seconds on average. Thus, this

strategy would suggest that useful threshold values may be found in the range of 3.6 (=1.8+1.8) to 7.0 (=3.0+4.0) seconds.

Table 1 Calculations of task execution times, according to the Keystroke Level Model (Card, Moran and Newell 1990)

	Time (in sec)	Activity
T(mental)	1.35	Mentally prepare to do something
T(home)	0.40	Homing hands on the keyboard or other device
T(key)	0.08	Best typist (135 words per minute)
	0.50	Typing random letters
	1.20	Worst typist (unfamiliar with the keyboard)
T(response)	t	System response time (this time is only calculated as far as it exceeds Tmental – it is assumed that while waiting for the response the user can mentally prepare for the next action)
T(execution)	min 1.83+t max 2.95+t	=Tm+Th+Tk+Tr (=Total task execution time)

The estimates made with KLM are based on extensive empirical measurements of actual users. However, all users were computer users, performing text-editing tasks. Newell's (1990) Unified Theories of Cognition (UTC) was used to provide "order of magnitude" estimates, based on different information. UTC works with "time scales for human action", which are derived from more general findings on human information processing. According to UTC, Card, Moran and Newell's *unit tasks* typically take in the order of 10 –100 seconds. It is assumed that unit tasks are composed of multiple *simple operations*. For *simple operations* (as Newell defines it) "*responses must be known and simple (press a button, vocalize a word) ... the connection of the response with the eliciting situation (the stimulus) must be clear to the subject*" (Newell 1990). The time scale for *simple operations* is typically in the order of 1-10 seconds. This seems to be in line with the estimations derived from the KLM.

As a preliminary procedure for DEVAN it is suggested that analysts determine threshold pause times on an ad hoc basis. For each set of user tests, analysts should conduct a pilot data analysis to derive useful (fixed) threshold pause time values (see section 3.3.1 for more details). The estimates made with the KLM and UTC can serve as rough indications for the order of magnitude to be used. In section four, this preliminary procedure is evaluated, and a more explicit procedure is proposed.

3.1.2 What format is used for the interaction overview?

The format for representing the interaction (from now on 'interaction table') uses three central columns that (side-by-side) represent the interaction at (minimally) three levels of abstraction (see figure 1). The leftmost of these columns (figure 1, item 1) consists of a simple log of all actions that a user performs on the product (with time stamps and information on the product's status after the action). In addition, this column is used for a preliminary segmentation of the interaction, based on pause times

What's the problem?

Studies on identifying usability problems in user tests

only. For this, DEVAN makes use of primary and secondary threshold times (figure 1, items 2 and 3; further explanation on primary and secondary threshold times can be found in section 3.3.1). Primary segment boundaries are represented as thick lines between actions; secondary boundaries are represented as dotted lines.

The middle column of the interaction table shows the definitive clustering of actions to segments (figure 1, item 4). This clustering is based on the preliminary segmentation shown in the leftmost column, combined with analyst judgments (for the exact procedure, see section 3.3.1). Short phrases characterizing the interaction segments are shown in the middle column as well. These phrases make explicit what the analyst thinks is happening within that segment. They may take the form of supposed user goals (e.g., “*user wants to set the scheduled time for program period leave*”), or may just be denominators for a group of actions (e.g., “*user scrolls through the picture menu*”).

The third column represents performance at the level of tasks provided to the test subjects (figure 1, item 5). The start of a new task is indicated by a thick line. Just below the line, task number and task description are added. If at some points in the interaction, analysts think that, for a better overview, segments should be clustered to intermediate level episodes, these can be represented in the third column as well (not shown in figure 1). Graphically, intermediate level episodes can be distinguished by using different font types and sizes. Analysts can add descriptions indicating what the clustered segments have in common. Again, descriptions may take the form of ‘user goal descriptions’ (e.g., if analysts feel they are reasonably sure towards what goal the user is working) or of denominators, merely describing what happens in these segments (e.g., *user scrolls the same menu multiple times, each time entering the same values for a menu-item*). Figure 2 shows part of an example interaction table.

3.2 How can usability problems be recognized?

After creating the interaction table (based on watching the videotapes several times), the interaction is studied in detail to locate usability problems (for more details on the use of videotapes during the analysis, see section 4.5.1). For this, DEVAN uses a list of event types each of which indicates the occurrence of a problem in the interaction. The list is based on Lavery, Cockton and Atkinson's (1997) definition of *usability problems*, which describes behavioral and performance consequences of usability problems in general. Furthermore, the list is derived largely from a taxonomy of problem phenotypes developed by Sutcliffe et al. (2000).

3.2.1 Theoretical considerations

Lavery, Cockton and Atkinson (1997) define usability problems as follows:

“A usability problem is an aspect of the system and/or a demand on the user, which makes it unpleasant, inefficient, onerous or impossible for the user to achieve their goals in typical usage situations.”

Time code	Action	Product status after action	Interaction segments	Task context	Breakdown indication code
			initiate entering current day and time	Task 1a Set clocktime to 7.10 and day to Tuesday	
15:49	<enter current day and time>	13:00 Monday			
16:00	<day>	Tuesday	select day		
16:00	<day>	Wednesday			
16:01	<day>	Thursday			
16:06	<time forward>	13:01	set time		act
16:06-16:08	<time forward>	13:05			act (rep?)
16:08	<time backward>	13:04			corr
(16:09)	(<time backward> failed)				exe
16:10-16:24	<time backward>	7:07			exe
16:25	<time forward>	7:08		corr	
16:26	<time forward>	7:09		corr (rep?)	
16:27	<time forward>	7:10		corr (rep?)	
	[reads task]				
			fix current day and time "fix"		
16:42	<enter current day and time>		afterwards: "now it is entered, I think"	goal act	

Figure 2 Example of an interaction table, taken from the thermostat test (see section 4.1.1), task 1a (translated from Dutch)

They distinguish between four components for any given usability problem:

"a cause, a possible breakdown in the user's interaction, and an outcome, all of which happen in a context."

In other words: within a specific context (e.g., user context, interaction context, task context), some cause (e.g., a design fault), may lead to a breakdown in the interaction (e.g., the user selecting an inappropriate action). This in turn may result in some undesired outcome (in terms of behavior and/or performance; e.g., the user's task fails, the quality of the work suffers, the user becomes irritated). In another paper, Lavery and Cockton (1997) make a distinction between dialogue failures and knowledge mismatches. Dialogue failures equal breakdowns as defined above. Knowledge mismatches are defined as occurring when the user does not have the correct understanding of a particular aspect of the system or task. In case of DEVAN, the word 'breakdown' will be used to include dialogue failures as well as knowledge mismatches. The words *unpleasant*, *inefficient*, *onerous* and *impossible* in Lavery, Cockton and Atkinson's (1997) definition refer to the outcome component of a usability problem, they refer to consequences of interaction breakdowns. Analysts can come to know about the

What's the problem?

Studies on identifying usability problems in user tests

existence of such consequences by listening to users (e.g., to their verbal utterances), by watching them (e.g., attending to non-verbal behavior like frowning) or by logging task performance (e.g., the user selecting a wrong action and then undoing it). Thus, at different points in time there can be multiple indications for the occurrence of a single breakdown. For example, an analyst may see that the user erroneously selects a button (first indication), may hear the user say “*oops, that was wrong*” (second indication) and may then see that the user undoes the erroneous action (third indication).

DEVAN makes use of a checklist to assist in systematically searching for such breakdown indications. The checklist is largely derived from the *problem phenotypes* taxonomy of Sutcliffe et al.'s Model Mismatch Analysis method (MMA) (Sutcliffe et al. 2000) which is based on a cyclic task-action model inspired by Norman's well-known model of action (Norman 1986). The model distinguishes between several interaction stages, including error correction and non-goal directed exploration. The concept of *problem phenotypes* is defined by Hollnagel (1993) as referring to “*how [problems] appear in overt action, how they can be observed, hence the empirical basis for their classification*”. It should be stressed that contrary to what Hollnagel suggests, the checklist in DEVAN is used solely for *locating* breakdown indications, and not for *classifying* them.

A number of adaptations were made to Sutcliffe et al.'s taxonomy. There were several reasons for this:

- 1 Some of the items in the taxonomy contained more than one indication at once. For example, one of the items in the taxonomy is ‘Puzzled, can not proceed’. Of course a user can feel puzzled and may not know how to proceed, but what if the user does not proceed and does not look puzzled at all? Should this be regarded as a sign that something is wrong? To avoid such confusion, DEVAN makes no use of combined indications.
- 2 In some cases, it was not clear what signals analysts had to search for, in order to label an event as ‘indication for a breakdown’. For example in the case of ‘Puzzled, can not proceed’: how can analysts decide that the user *can* not proceed (as opposed to *does* not proceed)? They can see a user starting a new task, quitting a session or they can *hear* them say that they *don't know* how to proceed (and then *infer* that indeed they can not proceed). DEVAN's checklist tries to be clear on the type of behavior that can count as evidence for the occurrence of a breakdown.
- 3 Sutcliffe et al.'s *action stages model* classifies phenotypes according to various ‘action stage contexts’. Thus, in analyzing an event, analysts have to interpret in what action stage the event took place before it can appropriately be classified. In many cases, it is not clear in what action stage an event happens, whereas it can still be clear that some breakdown must have occurred. For example, if a user seems puzzled and has not yet initiated an action, how can an analyst distinguish between ‘Puzzled, can not proceed’ (in the context ‘start of a task, or sub-task’) or ‘Hesitates, does not proceed’ (in the context ‘action selected or initiated’)? As DEVAN's checklist is not meant for classification purposes, there is no need to make this distinction. On the other hand, it is very likely that the ‘action stages contexts’ have played an important theoretical role in achieving completeness for Sutcliffe et al.'s

taxonomy. To preserve this completeness, it was required that all items of Sutcliffe et al.'s taxonomy could be related to at least one of the items in DEVAN's checklist. There is one exception to this requirement. Sutcliffe et al.'s item 'System freezes or crashes' was considered more of a technical problem than a usability problem (although it will certainly be an unpleasant experience to the user). In addition, during the development of the checklist, it was found that two additional items had to be included in the checklist. These are the items GOAL and REC (see table 2).

3.2.2 Events that indicate the occurrence of a breakdown

Table 2 Definition of breakdown indication types (final list, after implementing the improvements described in section 4.1.2; translated from Dutch).

Breakdown indication types based on observed actions on the product

Code	Short description	Definition
ACT	wrong action	an action does not belong in the correct sequence of actions an action is omitted from the sequence an action within a sequence is replaced by another action actions within the sequence are performed in reversed order
DISC	discontinues action	user points at function as if to start executing it, but then does not user stops executing action, before it is finished
EXE	execution problem	execution of action not done correctly or optimally
REP	repeated action	an action is repeated with the same effect
CORR	corrective action	an action is corrected with a subsequent action (or sequence of actions) an action is undone
STOP	task stopped	starts new task, before having successfully finished the current task

Breakdown indication types based on verbal utterances or on non-verbal user behaviour

Code	Short description	Definition
GOAL	wrong goal	user formulates a goal that cannot be achieved with the product or that does not contribute to achieving the task goal
PUZZ	puzzled	user indicates: not to know how to perform the task or what function is needed for it not to be sure whether a specific action is needed or not
RAND	random actions	user indicates: that the current action(s) are chosen randomly
SEARCH	Searches for function	user indicates: not being able to locate a specific function to be searching for a function of which the analyst knows it does not exist
DIFF	execution difficulty	user indicates: having physical problems in executing an action that executing the action is difficult or uncomfortable
DSF	doubt, surprise, frustration	user indicates: not to be sure whether an action was executed properly not to understand an action's effect to be surprised by an action's effect the effect of an action was unsatisfactory or frustrated the user
REC	Recognition of error or misunderstanding	user indicates: to recognise a preceding error to understand something previously not understood
QUIT	quits task	user indicates: to recognise that the current task was not finished successfully, but continues with a subsequent task

What's the problem?

Studies on identifying usability problems in user tests

Table 2 shows DEVAN's checklist of breakdown indication types. The list consists of two parts differing in what kind of behavior was observed: an action on the product or other user behavior (i.e. verbal utterances and non-verbal behavior).

To assist in keeping track of events that analysts interpreted as breakdown indications and to explain what the breakdown consists of, DEVAN makes use of a specific format for describing breakdown indications. Main requirements for this format are that

- based on the descriptions, analysts should be able to reconstruct what happened and why the observed events were regarded as indications for breakdowns;
- in subsequent analyses the descriptions should facilitate finding back the indications on tape (thus making the video contents more accessible).

The following elements are included in descriptions of breakdown indications:

- a time code reference, so that the event can be located on tape or in the interaction table;
- a description of the observed event;
- a description of the context in which the event occurred; i.e. the product mode in which it occurred, as well as the current task, intermediate-level episode and interaction segment;
- the code for the indication type (as specified by the checklist, table 2);
- a free-form description of the breakdown indication.

For additional reference, breakdown indication codes are added to the interaction table as well (see figure 1, item 6). They are shown directly to the right of the interaction table at the appropriate point in time. Furthermore, the event itself (action, verbal utterance, etc.) is marked grey in the interaction table (figure 1, item 7). See figure 2 for an example interaction table and table 3 for the related example list of breakdown indication descriptions.

3.3 What procedure is followed in doing the analysis?

To assist the analyst in creating the tool's various representations, DEVAN provides a procedural description of how to go about it in the data analysis process. Two main stages are distinguished, consisting of three and two sub stages respectively:

- 1 creating the interaction table:
 - *logging and transcribing* actions, verbal utterances and non-verbal behavior
 - *segmenting* the interaction based on threshold pause times
 - *clustering actions* to interaction segments, intermediate-level episodes and task-level episodes
- 2 creating the list of observed indications for breakdowns:
 - *locating* indications for breakdowns
 - *describing* indications for breakdowns

3.3.1 Stage one: Creating the interaction table

Before starting to make interaction tables for all subjects, threshold pause times have to be determined. This is done, based on the analysis of a pilot data set. After logging

Table 3 Example list of breakdown indication descriptions taken from the thermostat test (see section 4.1.1), task 1a (translated from Dutch). Breakdown indications between thick lines refer to the same breakdown.

Time code	Breakdown indication code	Free-form description	Observation	Product mode	Task context
16:06	act	uses <time forward> in stead of <time backwards>	presses <time forward>	enter current day and time	Task 1a: set clocktime to 7:10 and day to Tuesday; set time
16:06 - 16:08	act	uses <time forward> in stead of <time backwards>	presses <time forward>	enter current day and time	Task 1a: set clocktime to 7:10 and day to Tuesday; set time
16:08	corr	use of <time forward> is corrected with <time backward>	presses <time backward>	enter current day and time	Task 1a: set clocktime to 7:10 and day to Tuesday; set time
16:09	exe	does not press <time backward> hard enough	presses <time backward>, but display does not change	enter current day and time	Task 1a: set clocktime to 7:10 and day to Tuesday; set time
16:10 - 16:25	exe	overshoot while using <time backward>	presses <time backward>	enter current day and time	Task 1a: set clocktime to 7:10 and day to Tuesday; set time
16:25	corr	corrects overshoot of <time backward> with <time forward>	presses <time forward>	enter current day and time	Task 1a: set clocktime to 7:10 and day to Tuesday; set time
16:26 and 16:27	corr	corrects overshoot of <time backward> with <time forward>	presses <time forward>	enter current day and time	Task 1a: set clocktime to 7:10 and day to Tuesday; set time
16:42	goal	indicates verbally that he wants to confirm the newly made settings	prior to the action he says: "fix", afterwards he says: "now it is entered, I think"	enter current day and time	Task 1a: set clocktime to 7:10 and day to Tuesday; fix current day and time
16:42	act	presses <enter current day and time> after entering day and time (superfluous action)	presses <enter current day and time>	enter current day and time	Task 1a: set clocktime to 7:10 and day to Tuesday; fix current day and time

and transcribing the interaction of a pilot subject, the analyst intuitively chooses a candidate threshold time and uses this to segment the interaction. Subsequently, the resulting segmentation is evaluated for its 'meaningfulness'. In this context, 'meaningful' means that most of the segments seem to reflect "*well-integrated, coherent segments of behavior*", that relate to the syntactical structure of the product's interaction language. If almost all resulting segments are considered meaningful, the threshold is chosen as the primary threshold time. For additional structuring, a (shorter) secondary threshold can be used. After choosing threshold times, the analysis can start and the same threshold times are used for all subjects of a test.

The 'interaction table' stage starts with *logging and transcribing*. In this sub stage, all user actions, verbal utterances and obvious non-verbal behavior are logged, along with time codes and information on product status. This may be done using specific logging software, or it may be done by hand. Thus, this sub stage concerns 'filling' the leftmost column of the interaction table, as well as part of the middle column (i.e. transcribing verbal utterances and non-verbal behavior).

What's the problem?

Studies on identifying usability problems in user tests

In the second sub stage the interaction is *segmented*. This is done by going through the time codes and calculating pause times that exceed the primary or secondary threshold pause times. Thus, preliminary segments are formed, which are represented in the leftmost column of the interaction table.

The next step is the *clustering* sub stage. Clustering starts, based on the preliminary segments created in the previous sub stage as well as on analyst judgments. The procedure is as follows. All thick lines of the leftmost column (indicating the crossing of primary thresholds) are extrapolated to the middle column. In addition, dotted lines (indicating the crossing of secondary thresholds) may be extrapolated to thick lines in the middle column, as well. Analysts can decide to do so if they think it provides a useful additional way of structuring at that point. If, at places between thick or dotted lines, users explicitly state (verbally) that they start 'something new', analysts can decide to start new segments at such points as well. In addition to this extrapolation of lines, analysts enter short textual characterizations of the newly formed interaction segments in the middle column.

At this stage, higher level episodes may be defined as well. These are represented in the rightmost column of the interaction table. Boundaries of the highest level episodes are formed by start and finish of tasks. Task numbers and task descriptions are added to each high-level episode. If it seems useful, intermediate-level episodes may be shown in the rightmost column as well. If this is done, it is advised to create these in a data-driven way (i.e. by clustering interaction segments, rather than by theoretically subdividing tasks into subtasks). Hints for cases in which it is often useful to start new intermediate-level episodes are when:

- the user seems to start a new task
- the user seems to start a distinct new part of the task (i.e. a new cluster of related interaction segments)
- a distinct part of the task is repeated (i.e. another trial, or entering a subsequent set of values)
- a user stops working directly towards a task goal, but instead starts exploring functions just to learn something about the product or about previously made settings
- a user resumes the performance of an abandoned part of a task

At the end of stage one, the interaction table looks like that in figure 2 (except for the grey marks and the breakdown indication codes, which are added in stage two).

3.3.2 Stage two: creating the list of 'observed indications for breakdowns'

The '*breakdown indications*' stage starts with the sub stage of *locating* indications. At this sub stage, DEVAN's checklist is used to carefully study the interaction for locating events that indicate breakdowns in the interaction. To start with, the indications found are marked grey in the interaction table. The code for the type of indication is added directly to the right of the interaction table at the appropriate point in time. It is advised not to judge severity of breakdowns but to write down each and every indication regardless how insignificant it may seem (this is done so, because in subsequent analyses, new perspectives might make seemingly irrelevant indications more relevant).

At the *description* sub stage, the list of breakdown indication descriptions is created. The procedure for this is to simply go through the interaction table and ‘one by one’ describe all indications according to the format described in section 3.2.2.

4 Evaluating the tool

DEVAN was evaluated in two steps. Firstly, it was applied to data from a user test on a programmable thermostat. Some modifications to the list of breakdown indication types proved to be necessary. After making the modifications, DEVAN was applied to two data sets from a user test on a TV-video combination. In all cases, DEVAN was used by two independently working analysts (the first two authors of the present article). The following section describes the user tests as well as the modifications made after applying the tool to the first data set. Examples are shown of interaction tables, as well as of breakdown indication descriptions.

4.1 Application of the tool to existing datasets

First, the thermostat user test is discussed (section 4.1.1). Subsequently, modifications to DEVAN based on using it for this test are discussed (section 4.1.2). In section 4.1.3, the TV-video test is described.

4.1.1 The thermostat test

A programmable home thermostat (Honeywell Chronotherm III, see figure 3) was tested by five subjects (Vermeeren 1999). To evaluate the use of DEVAN, the data from one of these subjects were used (a subject who’s task performance was not too extreme in terms of speed and breakdowns).

In the thermostat test, subjects were given 12 tasks on paper. They were told that it did not matter whether they would finish all twelve tasks, but they were asked to try and finish each task successfully (however, they were told that they could skip a task if, after some time, they thought they would not succeed). Sessions lasted about 20 to 25 minutes. Tasks were formulated as scenarios, in terms of user goals describing a desired behavior of the heating system. For example: *“You are going away on holiday and don’t find it necessary that the house is heated during that time. Make settings such that the house will not be heated during the holidays”*. As the thermostat was not connected to a real heating system, there was no ‘real-life’ feedback. The thermostat in itself was completely functional and indicated when it ‘tried’ to switch on the heating. None of the subjects had any previous experience with using a programmable thermostat. All sessions were recorded on video (subjects’ hands as well as an overall picture showing the subject sitting at the table).

What's the problem?

Studies on identifying usability problems in user tests

For segmenting the interaction, a primary threshold time of 5 seconds was chosen. The secondary threshold time was set at 3 seconds. Figure 2 and table 3 show examples of an interaction table and a list of breakdown indications, respectively.



Figure 3 Honeywell Chronotherm III thermostat

4.1.2 Improvements after the first application of the tool

While applying the tool to data from the thermostat test, some problems in the use of the list of breakdown indication types showed up, leading to modifications of the list. For example, some of the indication types were merged to one, because these types were easily confused and distinguishing them provided no additional value. This was the case for the indication types *doubt*, *surprise* and *frustration*, which were merged to *dsf* (*doubt-surprise-frustration*). Furthermore, the indication types *undo* and *correction* were merged to *correction*, as distinguishing between the two proved to be very difficult and not very informative. It was decided to treat *undo* as a special kind of *correction*. The indication type *unrelated actions* was removed from the list. After trying to apply it, it became clear that purely on the basis of observed actions, one can not determine whether actions are unrelated (as seen from the user's perspective). In specific cases, this could be determined based on verbal utterances, but for that the indication type *random actions* already existed. Finally, a new indication type was added to the list; it was labeled '*recognition of previous error or previous misunderstanding*'. During the analysis, this indication was spontaneously found several times and was considered relevant enough to be included in the list. In addition, definitions of some of the indication types were modified to make them clearer. Table 2 shows the final list of breakdown indication types.

4.1.3 The TV-video test

After implementing the changes mentioned in the previous section, the tool was applied to data sets from a user test on another product. In that user test, a combined TV-video recorder (Philips type nr. 21PT351A/00, see figure 4) was tested with twelve subjects. Subjects sat behind a table with the remote control of the product, a user manual, a quick reference card and a TV-guide on it. Data sets of two subjects of age group 30 –40 years, were analyzed. One subject (user A) was a relatively quiet man, who worked in a reasonably systematic way. The other subject (user B) was a talkative woman who seemed to work less systematic and who experienced many problems in performing the tasks. Three VCRs with pre-recorded broadcasts from the channels Netherlands 1, 2 and 3 were connected to the antenna input, to simulate broadcasted channels. Teletext was not available. The TV-video combination was placed on a table, at a distance of about 4 meters from the subject. The experimenter sat next to the subject. Subjects were asked to perform 38 tasks in a maximum of one hour time. All tasks were related to the product's video functions and to general TV functions. They were read out loud to the subject. All sessions were recorded on video (showing the user's handling of the remote control as well as the TV in front view). Prior to the analysis, analysts used pilot data from another subject to arrive at useful threshold pause times. A value of 7 seconds was set as the primary threshold time and a value of 4 seconds as the secondary threshold time. Figure 5 and table 4 provide examples of an interaction table and of a list of breakdown indication descriptions, respectively.



Figure 4 Philips TV-video combination.

What's the problem?

Studies on identifying usability problems in user tests

In the following sections, the use of DEVAN is evaluated. Section 4.2 focuses on the procedure for choosing pause time thresholds. The use of breakdown indications for locating breakdowns is evaluated in section 4.3. In section 4.4 it is evaluated to what extent the tool has succeeded in making the data analysis process explicit. Finally, section 4.5 focuses on how much time was needed for the analysis.

4.2 Selecting thresholds for pause times

In section 3.1.1, it was stated that the procedure for choosing threshold pause times should be treated as preliminary. Characteristics of the preliminary procedure were that 1) threshold pause time values are chosen on an ad hoc basis using pilot data, 2) the tool distinguishes between primary and secondary threshold pause times and 3) the values for these threshold pause times might be found in the range of 1 to 10 seconds. In case of the thermostat test, the preliminary procedure led to threshold values of 5 and 3 seconds; for the TV-video user test, values were 7 and 4 seconds. Below, the procedure is evaluated in detail. Subsequently, a new, more explicit procedure is proposed.

Time code	Action	Product status after action	Interaction segments	Task context	Breakdown indication code
12:41	<pause/stop>	Pause, channel 3	Stop the video player to start watching TV	Task 13. Start watching the "Natte Neuzen-show" at channel 2	
12:47	<3>	Pause, channel 3	Switch to channel 3		(task problem, she forgot which channel she had to switch to)
12:49	<3>	Channel 33			act
12:50	<3>	Channel 3	"now this is channel 3 isn't it?"		corr
13:00	<rotary> clockwise	Channel 4	"oh no, 2" Switch to channel 2		exe
13:01	<rotary> counter clockwise	Channel 2		corr	

Figure 5 Example of an interaction table, taken from the TV-video test, user B task 13 (translated from Dutch).

Table 4 Example of a list of breakdown indication descriptions, taken from the TV-video test, user B, task 13 (translated from Dutch). Breakdown indications between thick lines refer to the same breakdown.

Time code	Breakdown indication code	Free-form description	Observation	Product mode	Task context
12:49	act	immediately after selecting channel 3 she (accidentally?) presses the same key (<3>) again, thereby selecting channel 33	presses <3>	Channel 3 is on (top right corner shows: 3-)	Task 13. Start watching the "Natte Neuzen-show" at channel 2; switch to channel 3.
12:50	corr	presses <3> again to return to channel 3 (from channel 33)	presses <3>	Channel 33 is on	Task 13. Start watching the "Natte Neuzen-show" at channel 2; switch to channel 3.
13:00	exe	turns rotary knob clockwise in stead of counterclockwise to go from channel 3 to 2	rotates rotary clockwise	Channel 3 is on	Task 13. Start watching the "Natte Neuzen-show" at channel 2; switch to channel 2.
13:01	corr	turns rotary knob counterclockwise to correct the channel setting from 4 to 2	rotates rotary counterclockwise	Channel 4 is on	Task 13. Start watching the "Natte Neuzen-show" at channel 2; switch to channel 2.

As a first step, it was investigated whether specific pause time values typically occurred more often than others. This was done by analyzing pause time frequency distributions for all three data sets. Figure 6 shows the three distributions. Per user test, fractions of the total number of observed pauses are depicted for each of the various pause time categories. Resolution for logging the actions was about 1 second. Therefore, the category of 0 seconds represents values of less than 1 second.

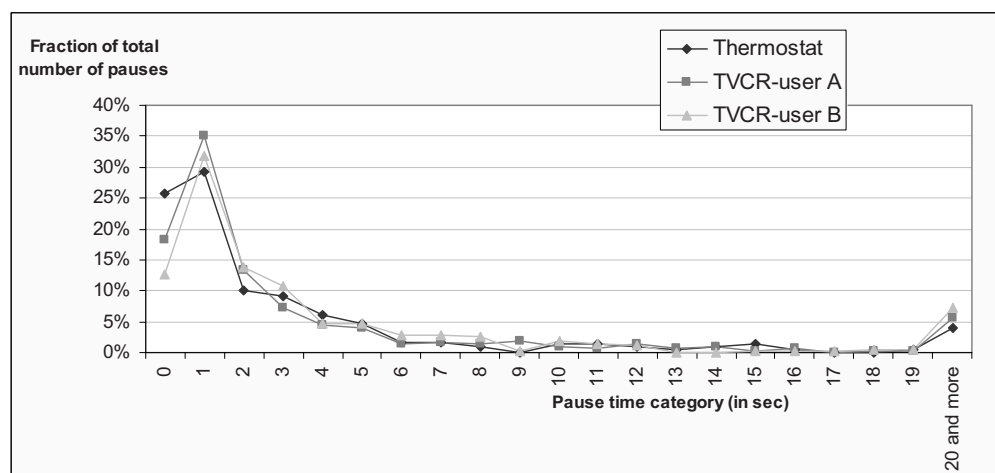


Figure 6 Frequency distributions of measured pause times for all three tests.

What's the problem?

Studies on identifying usability problems in user tests

All three distributions exhibit a 'peak' at '1 second'. As segmentation based on a threshold value of 1 second would largely result in segments consisting of single actions, this 'peak' has no meaning for determining useful threshold pause times. Apart from the peak at 1 seconds, all three distributions show a smooth, exponential decrease in pause times, the distributions seem to be almost identical. A two-tailed Kolmogorov-Smirnov test (Siegel and Castellan 1988) confirms this similarity. It shows that for the two subjects in the TV-video tests, distributions are not significantly different (at $p < 0.05$ level), and that the distribution of the thermostat data was not found to be different from that of user A of the TV-video test ($p < 0.05$). The distribution from the thermostat test proved to be significantly different from that of user B in the TV-video test ($p < 0.05$) (see table 5). However, at closer inspection, it seems that this difference is mainly caused by pause times in the category of 0 seconds. Such pause times have no meaning as thresholds, because using them would result in segments that equal individual actions. After excluding the '0 seconds' category from the data, pause time distributions showed no significant differences anymore (see table 6).

Table 5 Investigating similarity of frequency distributions.

Two-tailed Kolmogorov-Smirnov test for large samples	The largest discrepancy $D_{m,n}$ between the two cumulative distributions is:	Critical value for $\alpha = 0.05$	
thermostat vs. TV-video user A	0.0747 ($m=230, n=308$)	0.1185	no significant difference
thermostat vs. TV-video user B	0.1302 ($m=230, n=364$)	0.1146	significant difference
TV-video user A vs. TV-video user B	0.0874 ($m=308, n=364$)	0.1053	no significant difference

Table 6 Investigating similarity of frequency distributions, excluding the pause time value of 0 seconds.

Two-tailed Kolmogorov-Smirnov test for large samples	The largest discrepancy $D_{m,n}$ between the two cumulative distributions is:	Critical value for $\alpha = 0.05$	
thermostat vs. TV-video user A	0.0650 ($m=171, n=252$)	0.1347	no significant difference
thermostat vs. TV-video user B	0.0438 ($m=171, n=318$)	0.1290	no significant difference
TV-video user A vs. TV-video user B	0.0693 ($m=252, n=318$)	0.1147	no significant difference

It is concluded that pause time frequency distributions provide no indications for useful threshold pause times. Thus, a procedure based on a more detailed analysis of pause time patterns is needed. Such a procedure is in line with Card, Moran and Newell's (1990) statements, that "a chunk can be expected to reflect the syntactical structure of a

system's command language" and that *"in executing a method, the user is more likely to pause between chunks than within chunks"*. These statements imply that pause time patterns may indeed relate both to the syntactical structure of a specific interface, as well as to a specific person's interaction with the interface. For distinguishing (the shorter) pauses within chunks from (the longer) pauses preceding a chunk, one should study episodes displaying an interchange of these. Such interchanges typically occur in 'less problematic' interaction fragments.

Figures 7 and 8 (taken from the thermostat test data) illustrate how pause time patterns of not-problematic episodes indeed can reflect the distinction between longer and shorter pauses. These figures show similar task fragments from the beginning of a session and from halfway the same session. In both cases, temperature schedules are entered for a one-day period. Tasks are performed without errors. In case of this thermostat, settings are structured in a hierarchical fashion. Each (day) schedule consists of up to four periods.

For each of these periods, a starting time and a temperature can be set. Thus, a typical interaction sequence consists of first selecting the period to be scheduled, followed by setting the desired starting time and the desired temperature. This hierarchical structure is clearly recognizable in the pause time patterns shown in figures 7 and 8:

- pauses immediately preceding the selection of a new period (the black bars) are typically longer than pauses for subsequent (temperature and time) settings within that period (the grey and white bars, respectively);
- pauses immediately preceding the first 'temperature' (or 'time') setting for a period, are typically longer than subsequent 'temperature' settings (or 'time' settings respectively) for that same period.

Making diagrams similar to those of figure 7 and 8 might assist analysts in selecting useful threshold times in a more systematic way.

For the tests described in the present article, threshold pause time values were chosen using the preliminary procedure described in section 3.3.1. This means that after having watched the tapes of a pilot subject, threshold times were chosen intuitively and were then applied consistently.

It was then judged to what extent the resulting segmentation made sense or not. If necessary, threshold times were subsequently changed. Below, meaningfulness of the primary segment boundaries is evaluated again, now based on more explicit criteria. According to these criteria segment boundaries are considered meaningful if at the pause: 1) subjects made evaluative remarks, formulated new intentions or made other remarks indicating a shift of focus, or if 2) subjects were reading a task, a manual, a quick reference card, etc. If the above does not apply, a segment boundary can still be considered meaningful if it is defensible based on the interface's structure (for example, if after the pause the user starts using a different menu or setting a different parameter than before). Table 7 shows what proportions of the original primary segment boundaries were meaningful, according to the new, more explicit criteria. It shows high level of agreements between the judgments of the preliminary procedures and the new, more explicit criteria.

Table 7 Meaningfulness of primary segment boundaries re-evaluated using more explicit criteria. The table shows high level of agreements between the results from using the preliminary procedures and using the new, more explicit criteria. Thus, the procedure for judging meaningfulness of segment boundaries can be made more explicit, using the new criteria.

	Number of meaningful primary boundaries	
	according to preliminary procedure	according to the new criteria
thermostat	32	32
TV-video user A	21	24
TV-video user B	36	31

Based on these comparisons, it is concluded that instead of the preliminary procedure for defining threshold pause times (largely based on judgments), a new procedure using more explicit criteria is proposed (see next section).

4.2.1 Conclusions

Frequency distributions of pause times provided no indications for what values can best be used as threshold values (see figure 6). In addition, Kolmogorov-Smirnov tests indicated that pause time distributions did not significantly vary across subjects or test situations. Thus, it is not likely that frequency distributions or differences thereof can be used for choosing threshold pause times. It seems more useful to analyze what pause time patterns can be found in session parts in which users hardly experience any problems. Based on an evaluation of the meaningfulness of the used segment boundaries, a more explicit procedure for determining threshold pause times is proposed below. This procedure is as follows:

- 1 *Create bar graphs of pause times, using interaction fragments from pilot data.* Select session parts (from pilot data) in which a subject seems to perform tasks in an (almost) non-problematic way. For deciding on threshold values only those parts that are long enough to allow patterns in pause times to be revealed, should be used (i.e. seemingly coherent episodes of task performance, in which a substantial number of buttons are pressed). Create pause time bar graphs of the session parts, which are similar to those in figures 7 and 8.
- 2 *Select a candidate primary threshold value.* Based on the graphs, select a candidate primary threshold value. Expect useful values somewhere in the range of 2 to 10 seconds.
- 3 *Segment the entire (pilot) data file,* based on the chosen threshold value.
- 4 *Determine meaningfulness of the primary segment boundaries.* Applying threshold values for defining segment boundaries will always yield a number of boundaries that are not meaningful. As a first step:
 - Decide what proportion of primary boundaries should come up as meaningful, according to the criteria below.

What's the problem?

Studies on identifying usability problems in user tests

Determine meaningfulness of all segment boundaries by answering the following questions

- At each boundary: Are there any utterances indicating a change of focus, newly formed intentions or evaluations of previous actions, or is the subject reading the task, a manual, pieces of text on a display etc.
- In the absence of such indications: is a segment boundary at this point defensible based on the interface's syntax?

If the answer to one of these questions is "yes", the boundary is considered meaningful.

- 5 *Evaluate the number of meaningful segment boundaries.* Check whether the criterion for the required proportion of meaningful boundaries is met. For the tests described in the present paper, proportions proved to be 100% (thermostat), 96% (TV-video, user A) and 82% (TV-video, user B). Based on this, a requirement could be that at least 80% of all primary boundaries should be meaningful.
- 6 *Decide whether or not to use the chosen threshold value.* If the measured proportion is lower than required: try a different value (go back to step two), segment again and verify whether the new value fulfils the criterion better. If the proportion is higher than required, continue with the next step.
- 7 *Select a secondary threshold value.* Based on the bar graphs (see step one) and the chosen primary threshold value, select a secondary threshold value that is a few seconds lower than the primary one. During the analysis, this secondary value could suggest additional boundaries to the analyst. After selecting the secondary threshold the analysis can start.

4.3 The use of indications for breakdowns

One of the other key elements in DEVAN is the use of breakdown indications as a means of locating breakdowns. As stated in section 3.2.1 there can be various indications for a single breakdown. The checklist of breakdown indication types helps analysts in recognizing indications. In this section, it is investigated in how many cases breakdowns were located with multiple indications and whether there is a relation between the number of indications and the chance of locating the breakdown.

Figure 9 shows that in all three tests, more than one indication was found for about 30-40% of all breakdowns. Obviously, there is redundancy in the list of breakdown indications, but does this redundancy lead to a higher chance of recording breakdowns?

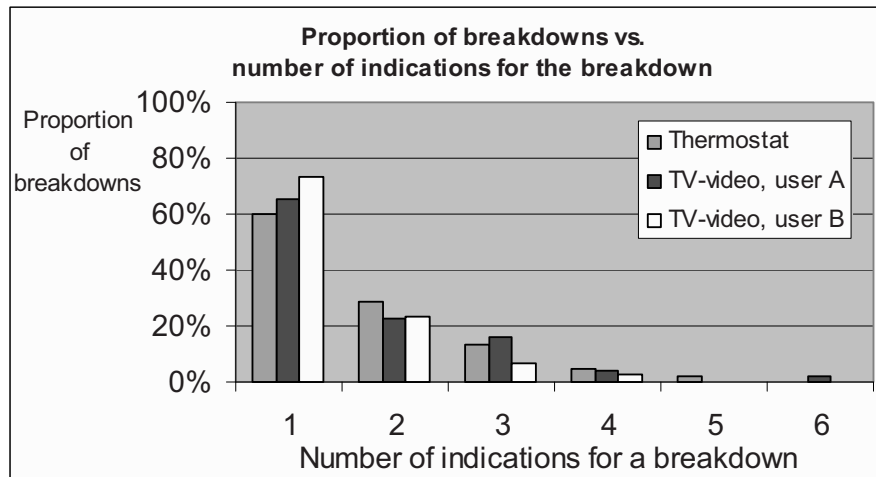


Figure 9 Numbers of indications found per breakdown. Total numbers of breakdowns: thermostat: 49, TV-video, user A: 54, TV-video user B: 110.

Tables 8, 9 and 10 show the number of breakdowns for which one, two or more than two indications were found, and relates this to the number of analysts who located the breakdown. The latter was used as an indication of the chance of recording the breakdown. For example, in table 8 it can be seen that 8 out of 9 breakdowns for which three or more indications were found, were collectively found by both analysts, whereas 1 out of the 9 was reported by only one analyst.

The figures in tables 8, 9 and 10 demonstrate a systematic effect. Firstly, if multiple indications were recorded for a breakdown, the breakdown was usually found by both analysts (thermostat: 18 out of 22, TV-video, user A: all 22, TV-video, user B: 30 out of 34). Secondly, if only one indication was recorded for a breakdown, then in only about half of the cases the breakdown was found by both analysts (thermostat: 12 out of 27, TV-video, user A: 21 out of 32, TV-video, user B: 36 out of 76). This suggests that a relation exists between the number of indications for a breakdown and the chance of recording it. Based on observed relations between the number of analysts that reported a breakdown and the number of indications for that breakdown one might even hypothesize that the number of observed indications for a breakdown can be used as a predictor of whether other analysts would report the same breakdown as well.

What's the problem?

Studies on identifying usability problems in user tests

Table 8 Relation between number of indications per breakdown and number of analysts that located the breakdown (Thermostat test).

Thermostat	breakdowns found by only one of two analysts	breakdowns collectively found by both analysts	total amount of breakdowns found
indication for the breakdown	15	12	27
indications for the breakdown	3	10	13
or more indications for the breakdown	1	8	9
total amount of breakdowns found	19	30	49

Table 9 Relation between number of indications per breakdown and number of analysts that located the breakdown (TV-video test, user A).

TV-video, user A	breakdowns found by only one of two analysts	breakdowns collectively found by both analysts	total amount of breakdowns found
1 indication for the breakdown	11	21	32
2 indications for the breakdown	0	11	11
3 or more indications for the breakdown	0	11	11
total amount of breakdowns found	11	43	54

Table 10 Relation between number of indications per breakdown and number of analysts that located the breakdown (TV-video test, user B).

Thermostat	breakdowns found by only one of two analysts	breakdowns collectively found by both analysts	total amount of breakdowns found
1 indication for the breakdown	40	36	76
2 indications for the breakdown	4	20	24
3 or more indications for the breakdown	0	10	10
total amount of breakdowns found	44	66	110

4.4 Explicitness of the data analysis procedure

Tables 8, 9, and 10, show that the amount of breakdowns that was found by only one of two analysts was substantial. For the thermostat it was 19 out of 49 breakdowns (39%), for user A of the TV-video it was 11 out of 54 (20%), and for user B 44 out of 110 (40%). This can be due to various reasons, including analysts having missed certain events during the observation, or differing in how to interpret a subject's behavior. In the latter case, analysts may even have disagreed on whether specific events should be regarded as breakdowns or not. As was mentioned in section 2.1, potential disagreements of analysts were anticipated in the development of the tool, being the very reason why decisions in the analysis have to be made explicit.

The explicitness of the decisions was evaluated by identifying breakdowns that were reported by only one of the analysts and then trying to trace back at which data analysis stage differences first occurred. This proved to be relatively easy to do. For only four out of the 74 breakdowns that were identified by no more than one analyst, could it not be explained why that was the case. Categorized by sub stage, table 11 shows the numbers of breakdowns that were identified by only one analyst.

Table 11 Differences in what breakdowns analysts located, traced back to the data analysis activity from which the differences originated.

	Number of times that the difference originated in the sub stage of:				
	logging actions	making transcriptions	interpreting user intentions	locating breakdown indications	describing breakdown indications
Thermostat	3 ^{*)}	6	5 ^{*)}	5	2
TV-video user A	0	0	0	11	0
TV-video user B	2	8 ^{**)}	14	22 ^{**)}	0
Total	3-5	12-14	17-19	36-38	2

^{*)} In two of these cases, it could not be determined whether the difference originated in the sub stage of logging or of interpreting intentions.

^{**)} In two of these cases, it could not be determined whether the difference originated in the sub stage of making transcriptions or of locating breakdown indications.

For 70 out of 74 breakdowns, differences could be traced back to specific data analysis activities.

- 1 *Differences in logging user actions.* Some of the differences in breakdowns were caused by differences in logging. In 4 out of 5 cases differences were caused by the fact that analysts did not exactly see whether a button was pressed or not. For example, in one case, an analyst missed a quickly corrected erroneous button press. Most likely, this was missed due to an inaccuracy in rewinding the tape. Differences like these are undesirable. Automated logging would have prevented the occurrence of 4 out of 5 differences in logging.

What's the problem?

Studies on identifying usability problems in user tests

- 2 *Differences in making transcriptions.* More than half of the differences in this category were caused by the fact that analysts sometimes did not report non-verbal behavior like frowning, visually scanning the interface, a finger floating above a button for some time, sighing, or browsing through the manual. Additionally, some differences were caused by not reporting evaluative remarks that were not directly related to a specific action (e.g., “it’s not functioning anymore”, “this thing is really difficult to use”). Another difference was caused by differences in hearing what a subject says. Differences like these are undesirable, but hard to avoid. For verbal utterances, one could agree to transcribe everything, but for non-verbal behavior it is hardly possible to define what has to be reported. Differences in transcriptions will always occur, but the tool makes them explicit for discussion amongst analysts.
- 3 *Differences in interpreting user intentions.* There were two main types of situations in which differences in interpreting user intentions occurred. The first type refers to situations in which analysts disagreed on whether they thought a user was actively searching for a button to continue task performance, or had decided to explore some part of the interface, and continue task performance later. The other type refers to situations in which analysts severely doubted what the user’s intentions were. In such cases, one analyst sometimes decided to report the guessed intention, whereas the other decided not to report it and just described what happened in the specific segment. Inherently, there is a lack of information about user intentions. Because afterwards user intentions can only be guessed, differences in this category are unavoidable. Based on just videotapes, there is no way to decide which guessed intention is correct and which is not, but the tool makes such differences explicit so that they can be discussed.
- 4 *Differences in locating breakdown indications.* This constituted about half of the differences. Four main categories of causes for differences were identified. The largest category consisted of situations in which subjects performed superfluous actions or applied very inefficient, but (in the end) effective strategies. Analysts sometimes disagreed on whether these superfluous or inefficient actions should be considered indications of breakdowns or not. There seems to be a ‘grey’ area for which opinions differ on whether such actions have to be considered problematic or not. DEVAN makes the analyst’s decision explicit so that it can be discussed. Another category was related to brief and vague utterances of which it was not always clear whether they had to be interpreted as indications of ‘puzzlement’ or not. In theory, one could try to eliminate this category of differences by better defining how ‘puzzlement’ can be observed. However, the authors’ experiences during the development of the tool are that this is hard to achieve. At least, the tool makes explicit how utterances were interpreted so that differences can be discussed. A third category of causes refers to situations in which subjects forgot some action and analysts did not notice. In all of these cases, analysts knew that these actions were necessary, but did not notice the missing action during the analysis. Such differences are hard to avoid. The last category consists of differences that occurred in situations in which (technically) all required settings were

made in a correct way, but one parameter was set to a wrong value (e.g., in case of the TV-video: the TV was set to the wrong channel, see figure 5 for an example). It is hard to decide in general whether such cases should be regarded as problematic or not. In some cases, it may be clear that subjects forgot what was asked in the given task (such problems are task problems instead of usability problems). In other cases, it can not be excluded that the error was attributable to a subject's misunderstanding of the interface. The only solution may be that analysts report all such indications.

- 5 *Differences in listing and describing breakdown indications.* In case of two breakdowns, analysts forgot to copy the breakdown indication codes from the interaction table to the list of breakdown indications.

4.4.1 Conclusions

Using interaction tables to analyze what caused the differences in the analysts' lists of breakdowns proved to be possible in most cases. In no more than four out of 74 cases, it was not possible to decide in what sub stage of the analysis a difference first occurred. Analyzing causes of differences in more detail led to the conclusion that only some differences could have been avoided. This refers to situations in which the occurrence of differences could have been prevented by: 1) automated logging, and 2) reporting all breakdown indications (even if they seem to be caused by a subject not properly remembering a given task). However, one should realize that the effect of these measures is expected to be small. In our case the number of differences between analysts would have been reduced from 74 to 65.

4.5 How much time did it take to do the analyses?

Detailed and explicit video data analysis is very time-consuming. Therefore, trade-offs have to be made between thoroughness of an analysis and time spent. DEVAN is meant for conducting thorough analyses and beforehand it was anticipated that using it would be time-consuming. For the analyses performed in both the thermostat test and the TV-video tests, data analysis times were recorded. Table 12 shows session times and analysis times for the three tests. Session time/analysis time ratios varied between 1:25 and 1:29. For detailed and explicit video analyses, high investments in time are largely unavoidable. Nevertheless, it seems worthwhile to analyze in what way the use of DEVAN could be made more efficient without compromising the quality of its results.

What's the problem?

Studies on identifying usability problems in user tests

Table 12 How much time (in minutes) did it take to do the analysis?

	thermostat	TV user A	TV user B
session time (in min)	20	30	40
analysis time (in min)	530	870	1005
ratio session time /analysis time	1:27	1:29	1:25

4.5.1 Evaluating the time needed for the analysis

Activities in all stages of the analysis were studied and it was evaluated how much time was needed for each of them. The analysis was done with two Hi8-videorecorders. All sessions were recorded on two videotapes (each showing a different view on the subject) and video recorders were operated manually. For creating the interaction tables and the lists of breakdown indication descriptions, Microsoft's Excel was used (see table 4 and figure 5 for examples).

In the sub stages of *logging* and *transcribing*, the analysis consisted of repeatedly going through a cycle of watching the video until some event occurred, then pausing it, slightly rewinding it to the exact moment where the event started, observing and listening to the subject, manually entering data on the computer (i.e. buttons pressed, video time codes, product status, verbal utterances, non-verbal behavior) and then continuing with a subsequent cycle until the end of a subject's session. Apart from watching the tape itself, the activities of winding and rewinding the tape, as well as manually entering the data proved to be very time-consuming. For *segmentation* of the interaction the video tape was hardly used, because it could be done almost entirely based on the logged data. In the sub stages of *clustering* and *locating breakdown indications* analysis cycles consisted of (re)winding the tape to a specific time code (i.e. searching for a specific fragment), watching the tape for some seconds, rewinding the tape to watch the segment again (often multiple times, to try and better understand what the subject was trying to do), and then entering data into the computer (i.e. drawing segment boundaries, describing interaction segments and entering the codes for breakdown indications). This cycle was only gone through for those parts that were difficult to understand. In this sub stage, activities like winding and rewinding the tape and entering the data required much time. In the sub stage of *describing breakdown indications* cycles consisted of gathering the information needed for the breakdown descriptions by reading the interaction table and then typing the information in the correct format. Most of the information was present in the interaction table, but was distributed across various cells. Therefore, copying, pasting and editing the information from the interaction table was experienced to be less efficient than re-entering it from a printed version of the interaction table.

4.5.2 Conclusions

Based on the analysis above it is concluded that an analysis with DEVAN can be made more efficient by using (partly) different and integrated hardware and software tools. More specific, a DEVAN analysis will be less time-consuming if

- log files can be used to control the video data, so that analysts no longer need to continuously switch between working with the computer and working with the video recorder
- log files or parts of log files are created automatically or if template-based logging software is available so that logging can be done by 'pointing-and-clicking' instead of by typing
- digital video is used so that winding and rewinding tapes is no longer needed
- dedicated software would exist to easily transfer and edit bits and pieces of information from the interaction table to a software template for breakdown indication descriptions.

Recently, data from another six subjects of the TV-video test were analyzed using Noldus' The Observer Video Pro (Noldus 2002) in combination with digital video stored on hard disc. This set-up made it possible to use log files for controlling video data, it provided a template-based way of logging and spending time on winding and rewinding tapes was no longer necessary. Informal time measurements from analyses of another six subjects of the TV-video test indicated that this new set-up leads to a reduction in time of about 30-40%.

5 General discussion

DEVAN was developed as a tool for structured and detailed analysis of video data from user tests of interactive systems. It is based largely on elements from earlier work on data analysis methods like SUPLEX (Cockton and Lavery 1999) and the Model Mismatch Analysis method (Sutcliffe et al. 2000). DEVAN was developed in the wider context of empirical studies on how the set-up of a user test may influence its outcomes. Such studies require careful, systematic and transparent data analysis processes, yielding complete and explicit results that can be discussed amongst researchers. Therefore, the aim for DEVAN was twofold. First, to structure the analysis so that all parts of the data receive due attention and, second, to make explicit the focus and perspective from which the analysis is done. The latter is accomplished by providing a general focus and perspective to take in the analysis (e.g., by defining what to log and what events to consider as indications of breakdowns) and by providing a framework that analysts can use to make their interpretations and decisions explicit (i.e. the interaction table format as well as the format for describing breakdown indications).

The use of DEVAN as a video data analysis tool was evaluated by having two people, independently analyze video data from three user test sessions. The data originated from user tests of a programmable home thermostat and of a TV-video combination. The use of DEVAN yielded detailed insights into how the analysts had interpreted the

What's the problem?

Studies on identifying usability problems in user tests

data: data analyses processes had been reported so explicitly that differences in the analysts' lists of breakdowns could easily be traced back to differences in specific data analysis activities. The results suggest that the majority of differences in the analysts' lists of breakdowns were caused by unavoidable differences in interpretations of subjects' behavior and that only minor improvements should be expected by refining the tool.

5.1 Future work

DEVAN was developed in a context which focused on consumer electronic products and interactions that typically consist of pressing labeled buttons and reading displays. Recorded data typically consisted of video and audio recordings with subjects performing tasks that were provided by an experimenter. Future research may focus on investigating how DEVAN can be used (or extended for use) in situations that may differ from the tested situations in the following ways:

Different interaction styles: Not all interactions are based on pressing labeled buttons and reading displays. For example, interactions may be based on dragging virtual objects on a screen, on physical manipulation of three dimensional objects or gestures, or on talking to a speech recognition system (e.g., see Preece, Rogers and Sharp 2002). In DEVAN, variations in interaction styles will be reflected mainly in modifications to the interaction table's logging column. For some interaction styles, a more visual format may be required, next to (or in stead of) the textual format that is currently used. Additionally, differences in interaction styles might have consequences for the segmentation procedure and for the list of breakdown interaction types. Further investigation is needed to study the extent to which this is the case.

Definitions of usability: There is a growing tendency to include criteria like 'fun' or 'pleasure' in definitions of usability (e.g., Jordan 2000). Especially in case of children's products such criteria seem to be of major importance (Oosterholt, Kusano and de Vries, 1996). For DEVAN, changes in the definition of usability will be reflected in changes to the list of breakdown indication types. For example, inclusion of 'pleasure' in the definition of usability would imply that the list of breakdown indication types should indicate how 'breakdowns in pleasure' can be observed based on actions, verbal utterances or non-verbal behavior.

Different types of test data: DEVAN's interaction table format, as well as it's list of breakdown indication types is based on logging actions, recording verbal utterances and observing a user's general behavior (gestures, facial expressions, etc). However, in methodological studies on user testing other data like eye movements may be gathered as well (e.g., Crowe and Narayanan, 2000). Such data could possibly be accommodated for by extending the interaction table's logging column with a column for eye-tracking data. However, one could also treat such data in a similar way as verbal utterances and incorporate them as annotations in the segments column. Additionally, such data might provide new indications for breakdowns, leading to an extension of the list of breakdown indication types.

Different techniques of user testing: DEVAN originates from a context of tests with single users and an experimenter that has only limited (verbal) interaction with those users. However, other user test techniques may require couples of subjects to collaborate in a test session (Kemp and van Gelderen 1996) or an experimenter assuming a more active role during the session (e.g., more conversation between subject and experimenter, or asking questions between tasks (Barnum 2002). For the use of subject couples the interaction table format will have to be modified so that each subject's contributions to the interaction can be separately logged. Discussions amongst subjects or between subjects and experimenters may provide an additional source of breakdown indications leading to an extension of the list of breakdown indication types.

Contexts where less time is available for the analysis: In analyzing user test outcomes a balance has to be found between a careful, detailed and explicit data analysis on the one hand, and time spent on doing the analysis on the other. For DEVAN, the balance was clearly tipped towards a careful, detailed and explicit analysis. Especially the explicitness of the data analysis process makes it very time-consuming to use. It would be interesting to investigate to what extent using separate elements of DEVAN could be useful as well. For example, one could consider doing a less explicit data analysis without creating interaction tables, but still using the list of breakdown indication types for locating breakdowns. In the study reported by Jacobsen (1999) a similar approach was taken to study methodological aspects of usability evaluation methods. Session time/analysis time ratios he found varied between 1:2.7 and 1:4.5. With DEVAN ratios varied between 1:25 and 1:29 (which could be reduced to about 1:15 to 1:20, see section 4.5). However, one should realize that tracing back causes of breakdowns is more difficult with less explicit data analysis techniques.

What's the problem?

Studies on identifying usability problems in user tests

chapter 3 Usability problem reports for comparative studies: consistency and inspectability

abstract

This study explores issues of consistency and inspectability in usability test data analysis processes and reports. Problem reports resulting from usability tests performed by three professional usability labs in three different countries are compared. Each of the labs conducted a usability test on the same product, applying a test protocol that was collaboratively developed by the labs. Each lab first analyzed their own findings as they always do in their regular professional practice. A few weeks later, they again analyzed their findings but then everyone applied the same method (SlimDEVAN: a simplified version of DEVAN, a method developed for facilitating comparison of findings from usability tests in an academic setting). It was found that levels of agreement between labs did not improve when they all used SlimDEVAN, suggesting that there was inherent subjectivity in their analyses. It was found that consistency of single analyst teams varied considerably and that a method like SlimDEVAN can help in making the analysis process and findings more inspectable. Inspectability is helpful in comparative studies based on identified usability problems because it allows for tracing back findings to original observations, as well as for laying bare the subjective parts of the data analysis.

This chapter was published as:

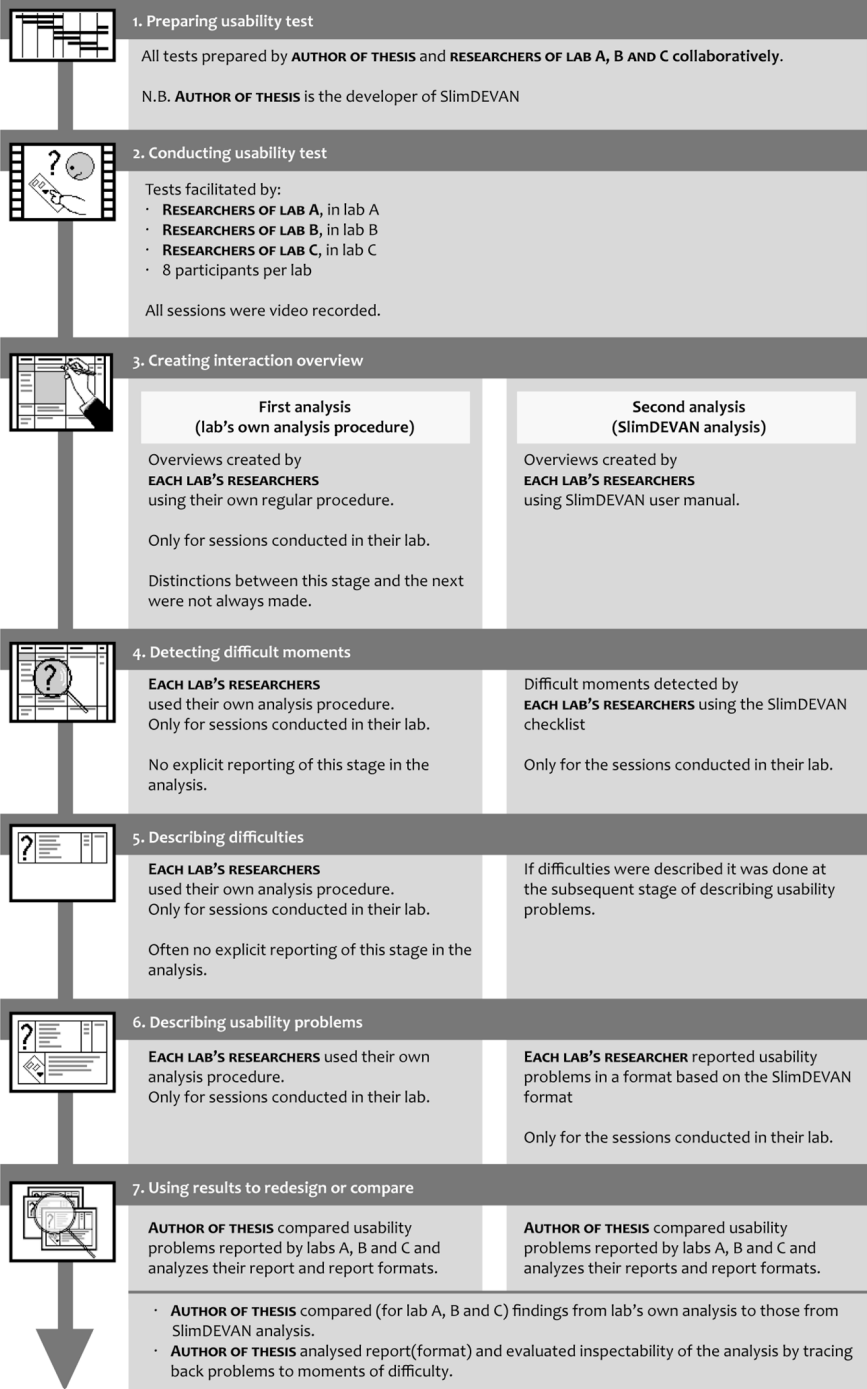
Vermeeren A.P.O.S., Attema J., Akar E., Ridder H. de, Doorn A.J. van, Erbug C., Berkman A.E., Maguire M.C. (2008) Usability Problem Reports for Comparative Studies: Consistency and Inspectability. In *Human-Computer Interaction* 23 (4), 329-380.

What's the problem?

Studies on identifying usability problems in user tests

Overview of characteristics of the studies reported in chapter 3

Product
· Digital oven interface panel



1 Introduction

Usability testing often takes place in the context of product development processes (for software, websites, electronic products, etc.) as a way of getting feedback on product usability. In some specific cases, findings from multiple usability tests need to be systematically compared. For example, in academic settings comparative studies may be conducted to study methodological issues of user evaluations (e.g., Jacobsen, Hertzum and John, 1998; Vermeeren, 1999; Cockton, Lavery and Woolrych, 2002; Molich, Ede, Kaasgaard and Karyukin, 2004; Vermeeren, Bekker, van Kesteren and de Ridder, 2007); in other settings they may be conducted to compare usability of competing designs or design proposals (e.g., Park and Lim, 1999; Hoenderdos, Vermeeren, Bekker, and Pierik, 2002) and in international product development settings they may be conducted to determine cultural differences in product use or usability (e.g., Daams and Hariandja, 2006; Noiwan and Norcio, 2006).

For being able to properly draw conclusions based on comparisons of findings from different test conditions, one needs to be reasonably sure that identified differences in findings can be attributed to differences in conditions, rather than to for example inconsistencies in data analysis or idiosyncratic focus or interpretations of an evaluator. Issues like these are often referred to as issues of reliability and subjectivity (or objectivity) respectively. In the present study, three labs in three different countries conducted usability tests on the same product, applying an agreed test protocol and then (independent from each other) analyzed their data and reported their findings. Based on the labs' problem reports, issues of reliability and subjectivity are studied with a focus on the usability labs' data analyses and reporting of results. Before proceeding to the introduction of the case study, the following section will discuss issues of reliability and subjectivity in more depth.

What's the problem?

Studies on identifying usability problems in user tests

1.1 Consistency of individual analysts (reliability)

Hartson, Andre and Williges (2001) discuss a number of quality criteria for Usability Evaluation Methods (UEMs) including: thoroughness, validity, effectiveness, and reliability. They define how to measure thoroughness, validity and effectiveness based on counting usability problems. However, they do not precisely define the concept of reliability in terms of how to calculate it; Hartson, Andre and Williges (2001) define reliability primarily in terms of evaluator agreement as "... a measure of the consistency of usability testing results across different users of the UEMs (evaluators)" (p. 396) However, they also speak about individual reliability, stating that "... In most UEMs, low individual reliability means high variability among evaluators, which means that merging results over a group of evaluators will give higher overall thoroughness" (p. 397). Thus individual reliability seems to refer to evaluations conducted by one single evaluator; presumably they refer to consistency of UEM results in cases where an evaluator applies a UEM multiple times on the same material (i.e., within-evaluator consistency).

In line with Hartson, Andre and Williges' definition, Guba and Lincoln (1989) state that (in research starting from a positivistic paradigm) the establishment of reliability "... typically rests on replication, assuming that every repetition of the same, or equivalent, instruments to the same phenomena will yield similar measurements" (p. 235). However, they also state that in research based on a naturalistic research paradigm, where by definition measurements cannot be exactly repeated, the issue of reliability (or dependability, as it is often referred to in that context) is dealt with mainly by making sure that the used process is "... an established, trackable, and documentable process," so that outside reviewers "... can explore the process, judge the decisions that were made, and understand what salient factors in the context led the evaluator to the decisions and interpretations made (Guba and Lincoln, 1989, p. 242)."

Kanis (1993) performed an extensive literature study to determine how the term reliability is used in two important constituent disciplines of human factors and ergonomics, namely technical sciences and social sciences. As a result of his inquiry he proposed "... to maintain a clear-cut distinction between random variation and systematic deviance in outcomes of measurements" (p. 96) and to use the term reproducibility rather than reliability. According to Kanis (1993), reliability (or reproducibility) should deal with random variation, rather than with systematic deviance.

In case of usability test data analyses based on extracting usability problems, the analyst forms part of the instrument for identifying problems; after all, ultimately, it is the analyst who judges whether an event is considered problematic or not. Presumably, analyst judgments are largely based on individual expertise, experiences, and ability to empathize with users. In such a context reliability concepts based on 'repeated measurements' are problematic as they assume that in the repeated analysis, the analyst has forgotten everything about the previous analysis, and has not gained any relevant, additional knowledge or experience affecting his/her perception of interactions (which will probably never be completely true). For the same reason it is also questionable to

what extent differences in findings from multiple analyses of a single analyst can be regarded as 'random variation' (cf. Kanis, 1993). Therefore, in the remainder of this article the term 'reliability' will be avoided and the issue will be dealt with mainly in terms of 'consistency of individual analysts'.

In this article, consistency of individual analysts (analyst teams) will be dealt with by using the repeated measures approach (*within-team consistency*), but taking into account Guba and Lincoln's (1989) advice to use established, documentable and traceable processes.

1.2 Consistency of findings across multiple analysts (subjectivity)

According to Guba and Lincoln (1989) objectivity is "*... concerned with assuring that data, interpretations and outcomes of inquiries are rooted in contexts and persons apart from the evaluator and are not simply figments of the evaluator's imagination*" (p. 243). In addition, they state that in a naturalistic research paradigm one may not assume that methods can prevent the inquirer (even inadvertently) introducing subjectivity in findings. Instead, assurances of integrity of findings are rooted in the data themselves. In other words, the starting point is that (at least some degree of) subjectivity is acknowledged in data analysis and should be dealt with properly. In Guba and Lincoln's view this means that both the original data and the processes used to compress these data should be available to be inspected and confirmed by outside reviewers of the study. For this they adopt a criterion of confirmability rather than objectivity. Probably, the term *inspectability* would be more appropriate, as the products and processes should be available for *inspection* (not only for confirmation, but also for falsification).

Because in usability test data analyses based on extracting usability problems the analyst forms part of the measurement instrument, Hartson, Andre and Williges' (2001) definition of reliability as "*... a measure of the consistency of usability testing results across different users of the UEMs (evaluators)*" (p. 396) (and analyst agreement in general) should be seen as primarily dealing with issues of subjectivity/objectivity, rather than with reliability purely. Findings from a number of studies investigating the so-called evaluator effect seem to confirm this notion. The evaluator effect (e.g., Hertzum and Jacobsen, 2001; Vermeeren, van Kesteren and Bekker, 2003; Law and Hvannberg, 2004) is the effect that if several evaluators analyze the same video recorded user test sessions there is a considerable lack of overlap in their findings. The findings from the studies suggest that no matter how careful, structured and detailed the analysis is, if it involves qualitative analyses there is a considerable subjective component in it (e.g., see Vermeeren, van Kesteren and Bekker, 2003). This suggests that, also in case of user test data analyses, assurances for objectivity cannot be rooted entirely in the method used and its subjectivity will have to be dealt with somehow.

In the remainder of this article, issues of subjectivity and objectivity will be dealt with by using measures of agreement between analyst teams as well as by keeping in mind Guba and Lincoln's (1989) advice that it should always be possible to trace back on

What's the problem?

Studies on identifying usability problems in user tests

what data the findings are based and how the data transformed from primary observations into findings (i.e. findings should be *inspectable*).

1.3 Aim of the study

Aim of the study presented in this article is to investigate issues of consistency and inspectability of data analyses and reports from usability tests based on extracting usability problems. For that purpose usability tests were conducted by lab teams in three different countries. They all tested the same product, applying the same test protocol. Subsequently, each individual team analyzed their data and reported about it. After the teams had finished writing their reports, they were asked to re-analyze their data a few weeks later, now applying (a simplified version of) the DEVAN tool (Vermeeren, den Bouwmeester, Aasman and de Ridder, 2002); the DEVAN tool was originally developed for improving an analyst's consistency in data analysis, for documenting the analysis procedures and for making findings inspectable (in order to facilitate recognition of subjectivity in each of the data analysis stages).

The teams' reports formed the basis for making various comparisons. The first step to making comparisons was to compile a 'complete' master list of usability problems from the teams' reports. As problem formulations in team reports were sometimes too ambiguous or incomplete to understand problems in enough detail for direct comparison, there proved to be a need to track back reported problems to their original sources. Experiences in trying to do so, have provided insight into issues of inspectability of the teams' reported findings. Based on the master problem list, measures of consistency (within-team consistency, as well as agreement between teams) were calculated. Inconsistencies in findings were analyzed in more depth by trying to trace back findings to the original data (inspectability) and identifying possible causes of differences. Identified causes of differences indicate whether these are due to issues of inconsistency of individual analyst teams or of inconsistency between multiple analyst teams. Also, those procedures that had been described in enough detail and those findings that were inspectable enough, indicated at what stage in the data analysis process inconsistencies (either within individual analyst teams or between analyst teams) occurred. In the next section, the usability tests conducted by the three labs are described, along with the analyses the lab teams performed and the way they reported their findings. Subsequently, in section 3, the reference analysis procedure (the simplified version of the DEVAN procedure) is explained, along with the teams' report formats that resulted from applying it. Section 4 discusses the procedure, experiences and results of comparing the reported findings and it reports the identification of relevant analysis and report characteristics. In section 5 the results are discussed and implications for data analyses and usability problem reports in practice are drawn.

2 Case study

This section introduces the usability tests that the labs conducted. It then discusses the ways in which the lab teams analyzed the user test data according to their usual professional practices and shows examples of the report formats used.

2.1 Usability tests on an advanced oven interface

2.1.1 The laboratories

The laboratories that conducted the user tests were UTEST at the Middle East Technical University in Ankara (Turkey), the WIT-lab at Delft University of Technology in the Netherlands, and ESRI at Loughborough University in the United Kingdom.

At the time of the test, UTEST was a relatively new usability laboratory within Middle East Technical University. UTEST provides services to industry and promotes academic studies. The collaboration with industry consists of both research and consultancy activities embracing military and consumer products as well as software and electronic appliances. Evaluators sitting in the control room of the lab can observe users in the test room through monitors and a one-way mirror. Remote-controlled and portable digital cameras are used to record user performance and observational software is used for data analysis. The research team consists of experts having diverse academic backgrounds.

The WIT-lab (Laboratory for Work and Interaction Technology) is a laboratory at Delft University of Technology, for both commercial usability services and for the support of research and educational projects. At the time of the study it had more than ten years of experience in commercial usability services. The lab has two test rooms, an evaluation room and a control room. Staff members have a background in organizational psychology, experimental psychology and systems engineering.

ESRI (The Ergonomics and Safety Research Institute) is located within Loughborough University and was formed from two institutes on ergonomics and human factors (HUSAT and ICE) dating back to the early 1970s. ESRI provides research, consultancy and teaching in the area of human interaction with computer systems, products and advanced technology. The ESRI usability laboratory is fitted with audio-visual equipment for testing product usability and is split into two areas: a user-work area and an evaluator's control room from which users can be monitored directly through a one-way mirror. Remote-controlled and portable video cameras are used to capture the users' interactions and performance with the product. The camera images are mixed and stored on tape for analysis. ESRI staff has a background in ergonomics, computer science and psychology and has more than ten years of experience in offering usability services in a commercial context.

What's the problem?

Studies on identifying usability problems in user tests

2.1.2 The product

The product that was used as the object of evaluation was a prototype of an advanced oven interface panel. The interface consisted of a combination of an LCD display with two push buttons and a rotary knob; at all times during product use, the function of each button is shown on the display. In the test room a prototype of the complete oven casing was shown, with a non functioning interface on it. Next to it was a cardboard mockup with a functioning interface on it (see figure 1).



Figure 1 A participant interacting with a mockup of the oven's interface panel.

2.1.3 Sessions and task scenarios

Test protocols were collaboratively developed by the three lab teams. Scenarios were defined prescribing what tasks each participant had to perform with the control panel; these scenarios covered most of the functionality of the oven and were phrased as real-life contextual goals to be reached by participants; for example: "Suppose the test room is the kitchen of a friend of yours. You enter the kitchen and you see that it is filled with smoke and that the smoke is coming from the oven. You see that the oven is working... something is cooking inside. Please go in and try to stop the cooking." and "For some special dishes the oven knows how it has to be set. Now suppose that you want to grill a large sized fish. See if the oven knows this recipe and if it does then start grilling the fish." After each task scenario, follow-up questions were asked for clarification of actions with unclear intentions (e.g., "what did you expect would happen when you pressed that

button?" or "what did you think had happened when you saw that screen appear?"). On average, sessions lasted about one hour.

2.1.4 Participants

Sampling of participants was done according to the manufacturer's market profile. In each of the three countries eight participants took part in the trials (6 female and 2 male, age varying between 20 and 55 years). All participants were regular cooks and part of a family of 3 to 5 members. They all had either recently purchased an oven or had recently considered the possibility of purchasing one. Other characteristics included: they are full-time or part-time employees; they are indigenous individuals, they cook for themselves at least 4 times a week; they not always cook ready-made meals; they live in a city or town environment.

2.2 The lab specific data analysis procedures

For each of the labs, the analyst team's data analysis procedure is described below.

2.2.1 Team A

In lab A two researchers participated in the test. Researcher 1 acted as facilitator, conducting the test sessions (providing participants with tasks, helping them if necessary and asking questions). During task performance, researcher 2 (acting as observer) watched participants performing tasks, took some notes and sometimes discussed with two student observers what exactly was happening. After the sessions, researcher 1 (the facilitator) watched the video recorded task performance sessions and wrote notes about the sessions in a format as shown in Figure A1 (Appendix A). Researcher 2 (the observer) and the two student observers did not take part in the analysis of the video recorded sessions.

2.2.2 Team B

In lab B, two researchers were involved in the test. Researcher 1 (the observer) manually logged sessions in real time during task performance and interviews, using dedicated database software. Logs were automatically time-stamped and linked to the digital video recordings. Figure A2 (Appendix A) shows an example log (NB. The logs were not part of the reports that were handed in and thus could not be used in the comparisons). Researcher 2 acted as facilitator during the test, but did not take part in the data analysis.

After the sessions, researcher 1 went through the event logs (during the analysis video recordings were automatically kept synchronized with the log files) and assigned so called *findings* (key usability-related observation) to logged events. Findings were then categorized according to tasks. Subsequently, for each finding a paragraph discussing the *weight* of the findings was written, as well as a paragraph providing *suggestions* for solutions. Weight and suggestions for solutions were inspired by going through the additional *non-finding* events in the log files (e.g., a user's additional verbal comments).

What's the problem?

Studies on identifying usability problems in user tests

Finally, for each finding a number of example events was selected for inclusion in the report. This resulted in a report format as shown in Figure A3 (Appendix A).

2.2.3 Team C

In lab C, two researchers participated in the test: researcher 1 acted as facilitator, researcher 2 as observer. During task performance, researcher 2 (with no strong understanding of the product interface) took notes on a printed version of the task protocol, focusing mainly on timing of tasks, as well as on key comments and actions of participants. Researcher 1 took notes on a printed version of the task protocol, using their own defined abbreviations. The notes from both researchers were discussed amongst them and were then combined and typed up as a single record of each session. The data analysis was jointly done by both researchers. Subsequently, researcher 1 wrote a report based on the combined notes. Video tapes were now and then used as a reference during the process. Figure A4 (Appendix A) shows examples of team C's report.

3 The reference analyses

One to two months after the teams had reported their findings, the videotaped sessions were analyzed again. This time a prescribed, detailed analysis was performed, using SlimDEVAN (a simplified version of DEVAN (Vermeeren, den Bouwmeester, Aasman and de Ridder, 2002)) as a reference to compare the initial analysis to. Below, SlimDEVAN will first be explained, followed by a brief description of how the teams got acquainted with it. Then the teams' SlimDEVAN analyses will be presented together with the report formats they resulted in.

3.1 Description of SlimDEVAN

SlimDEVAN is a checklist-based approach to user test data analysis. It is a simplified version of the DEVAN technique for video data analysis (Vermeeren, den Bouwmeester, Aasman and de Ridder, 2002). Main differences between DEVAN and SlimDEVAN lie in the way in which overviews of interactions are made. In case of DEVAN, the procedure for arriving at the overviews as well as the format for the overviews are prescribed in much detail; in case of SlimDEVAN decisions on these issues are largely left to the individual analyst, but advice and constraints are given. Both DEVAN and SlimDEVAN make use of a checklist. The checklist (see figure 2) aids in detecting events that signal the existence of interaction difficulties by defining such events. In this context, the term *difficulty* does not necessarily refer to a complete halt in task performance. For example, hesitations before (or frustration after) successful task performance are also regarded as difficulties, as are erroneous actions that are corrected instantaneously. The use of the checklist stimulates that evaluators use the same definition of what constitutes an

interaction difficulty. Moreover, it makes the analysis process more explicit. The DEVAN checklist is based on Lavery, Cockton and Atkinson's (1997) definition of *usability problems* which describes, in general terms, the *behavioral* and *outcome* consequences of usability problems. The SlimDEVAN checklist is basically the same as the DEVAN checklist (Vermeeren, den Bouwmeester, Aasman and de Ridder, 2002) but was slightly adapted based on experiences in other projects (e.g., Barendregt and Bekker, 2006; Vermeeren, Bekker, van Kesteren and de Ridder, 2007).

Basically, two types of observations are distinguished within the checklist. These are:

- physical actions performed on the product (i.e. actions performed on the products' control elements);
- expressions: (verbal) utterances from users, as well as body language (i.e., facial expressions, gestures, etc.).

The checklist assumes that both types of difficulty signals can be found at several stages of performing an action: (a) *prior* to physically performing the action (e.g., user hesitates before acting), (b) *during* the physical performance of an action (e.g., mispressing a button), (c) *directly following* an action (e.g., exclamation of surprise after seeing the system's reaction to an action) or (d) *later in a session* (e.g., when after continuing with other actions the user suddenly notices a preceding erroneous action and corrects it).

The SlimDEVAN approach works best if the analysis starts from a session log that specifies time-stamped actions on the product, as well as (verbal, gestural, facial) user expressions. The procedure is to go through a log file (or if desired also review parts of video taped sessions) and search for the types of events as defined in the checklist (the so-called difficulty signals). Codes for detected difficulty signal events are added to the session logs (thereby making the events time-stamped). Subsequently, for each participant, a list of difficult moments is created, preferably with time-stamps added. A single moment of difficulty can be signaled by multiple event types at a time (i.e., a single usability problem can be identified based on multiple signaling events). Figure 3 shows examples of how moments of difficulties can be specified (these examples are taken from the SlimDEVAN user manual (Vermeeren, 2003)).

What's the problem?

Studies on identifying usability problems in user tests

Breakdown signals in the form of physical actions performed on the product

(code, short description: definition):

- ACT, wrong action:** An action does not belong in the correct sequence of actions, an action is omitted from the sequence, an action within the sequence is replaced by another action, or actions within the sequence are performed in reversed order
- DISC, discontinued action:** User points at function as if to start executing it, but then does not, or user stops executing action, before it is finished.
- EXE, execution problem:** Execution of action not done correctly or optimally.
- REP, repeated action:** An action is repeated with exactly the same effect.
- CORR, corrective action:** An action is corrected with a subsequent action (or sequence of actions), or an action is undone.
- STOP, task stopped:** User starts new task, before having successfully finished the current task.

Breakdown signals in the form of utterances (verbal, sound) or body language (facial expressions, gestures) (code, short description: definition):

- PER, perception problem:** User indicates (in words or behavior) not to be able to hear or see something clearly.
- INTN, wrong goal or intention:** User formulates a goal that cannot be achieved with the product or that does not contribute to achieving the task goal; or user (verbally) specifies an action that s/he think is needed in order to progress towards the goal, but the specified action is not correct (indicating wrong user intention).
- PUZZ, puzzled (before an action):** User indicates (in words or behavior) not to know how to perform the task or what action is needed for it, or not to be sure whether a specific action is needed or not.
- RAND, random actions:** User indicates (in words or behavior): that the current action(s) are chosen randomly
- SEARCH, searching for a function (but not finding it):** User indicates (in words or behavior): not being able to locate a specific function
- DIFF, execution difficulty:** User indicates (in words or behavior) having physical problems in executing an action, or that executing the action is difficult or uncomfortable
- DSF, doubt, surprise, frustration (after an action):** User indicates (in words or behavior) not to be sure whether an action was executed properly, not to understand an action's effect, to be surprised by an action's effect or that the effect of an action was unsatisfactory or frustrated the user.
- WEX, wrong explanation (after an action):** User formulates an explanation for something that happens, but this explanation is not correct; or user formulates an interpretation for displayed feedback, but this interpretation is not correct.
- REC, recognition of error or of misunderstanding:** User indicates (in words or behavior) to recognize a preceding error, or to understand something previously not understood
- QUIT, quits task:** User indicates (in words or behavior) to recognize that the current task was not finished successfully, but continues with a subsequent task; or user indicates (in words or behavior) that s/he thinks a task was successfully finished and continues with a subsequent task, (whereas in fact the task was not finished successfully).

Figure 2 The SlimDEVAN checklist (at the time of the test the word breakdown was used in the checklist, as a synonym for the word difficulty in the present article).

Time stamp and signal codes	Free-form breakdown description	Inferences about what design elements may have caused the breakdown to occur.
0:02:40 ACT DSF 0:04:20 CORR	User wants to change <i>Time style</i> , and clicks at the time in the bottom right corner. Apparently, he expects that time style settings can be found there. He should have gone to <i>regional settings</i> in the <i>control panel</i> .	At two places there are settings related to <i>time</i> . At only one of those places it is possible to change <i>Time Style</i> . User expects it to be where the clock is, but it isn't.
0:03:10 INTN ACT	User expects to find <i>Time style</i> settings in the <i>Time Zone</i> tab of the <i>Date/Time Properties</i> , instead of in the <i>Regional Settings</i> in the <i>Control Panel</i> .	At two places there are settings related to <i>time</i> . At only one of those places it is possible to change <i>Time Style</i> . User expects it to be where the clock is, but it isn't.

Figure 3 Example format for a usability problem list that was made available to the teams (at the time of the test the word breakdown was used as a synonym for the word difficulty).

3.2 How the teams learned to use SlimDEVAN

The first time the teams heard about DEVAN was during the first project meeting with all teams. In that meeting they were told about the existence of DEVAN and were provided with copies of the article in which the tool was first introduced (Vermeeren, den Bouwmeester, Aasman and de Ridder, 2002). At that time, however, DEVAN was not considered feasible for use in the project, due to its time-consuming nature. At a later stage, after the second project meeting, the idea of using SlimDEVAN arose.

A brief description of how SlimDEVAN differed from DEVAN was sent to all partners by email, along with a few paragraphs of information about what actions would be required from them if they would be willing to use SlimDEVAN. After the three teams agreed on using SlimDEVAN, a user manual (Vermeeren, 2003) was written. The user manual included a separate checklist reference card specifying the SlimDEVAN codes for difficulty signals. In a third meeting the teams were provided with a copy of the user manual. In addition, the procedures for using it were orally explained and questions were answered in order to clarify what the teams could expect while using it. It was suggested to the teams that they could send part of the results of the analysis of one participant's task performance to the first author of the present article (who had not been involved in conducting the tests). In this way, they would have the opportunity to have their analyses checked for misunderstanding SlimDEVAN. Team C made use of this possibility.

What's the problem?

Studies on identifying usability problems in user tests

3.3 The teams' SlimDEVAN analyses

Below, each team's implementation of the SlimDEVAN reference analysis is described.

3.3.1 Team A

In case of lab A, two researchers participated in the SlimDEVAN analysis. These were the same researchers as in the lab specific approach. First, the researchers together analyzed part of one participant's session that seemed difficult to analyze and discussed their implementation of SlimDEVAN. In this way they developed a common understanding about the use of it. Then, each researcher watched the videos of four participants and took notes using identical table formats (see Figure B1, Appendix B for an example). The checklist card was used as a reference during the analysis. In addition, researcher 2 (the observer) sometimes compared fragments from different sessions to ensure better consistency. Also, researcher 2 analyzed one (difficult to understand) session twice to feel more confident about the findings. Subsequently, researcher 2 went through all typed up tables of both researchers to check for consistency in assigning codes. This led to only a few changes in the tables of researcher 2. Finally, for each participant a usability problem list was created. This was done by researcher 1. Figure B2 (Appendix B) shows an example of the format used for reporting the problems.

3.3.2 Team B

In case of lab B, researcher 1 (the observer) did the analysis. The SlimDEVAN analysis started from the log files made in the team's initial analysis. Figure B3 (Appendix B) shows an example of part of a session log made by team B. Figure B4 (Appendix B) shows an example report format¹.

The researcher went through all log files on the computer and while going through them, the video recordings were automatically kept synchronized with the log files. The dedicated software for logging was modified to allow for entering SlimDEVAN codes as markers into the log files. An additional marker called *Comment* was added for difficulty signals found in the interviews and for comments from the researcher (e.g., ideas for solutions to problems). As the video recordings ran along with the log files, it was possible for the researcher to especially focus on those parts of the video for which no loggings were made, as well as on parts of the log files for which it was not clear what exactly had been observed. At several points the original log files (from the lab specific analysis) proved to be incomplete.

Subsequently, the log files (including the SlimDEVAN codes) were filtered such that a list of (SlimDEVAN) coded events was created. *Findings* were then defined based on the coded events and multiple events could be linked to a single finding. In the next step of the analysis, the findings were grouped into categories that emerged during the

¹ Although log files with SlimDEVAN codes were made for all eight participants, team B accidentally reported problems for only six participants.

process itself (e.g., consistency, changing parameters during cooking, etc.). Categories in turn were grouped into chapters. Summaries of the findings reported in a chapter were made and frequencies of findings were related to variables like participant, participant characteristics or tasks, to get more insight into those situations in which problems occurred (in the report this was referred to as *validity*). Descriptions of loosely judged problem severity were written as well as suggestions for solutions. In writing about validity, severity and suggestions, the researcher especially searched for inspiration by going through those events that had not resulted in *findings*.

3.3.3 Team C

In case of lab C, both researchers together watched the tape of one of the participants and discussed it in relation to the SlimDEVAN checklist. Researcher 2 then watched the video of that participant again and wrote down actions and times of actions. Subsequently, researcher 2 watched the video again to add participant expressions to the action log, as well as to assign SlimDEVAN codes to events. Researcher 1 followed the same procedure for the other seven participants. Notes were typed up by each of the researchers in a format as shown in Figure B5 (Appendix B). Finally, researcher 1 used the format as shown in Figure B6 (Appendix B) to create a list of usability problems and indicate which participants experienced each problem.

4 Comparing the reports

In this section, the protocols for making comparisons are presented along with the results of the comparisons. First, in section 4.1, how the findings in the teams' reports were re-formatted into a form that allowed for making comparisons will be discussed: across teams, as well as across each team's subsequent analyses. Issues of inspectability of reports as experienced in this process are discussed. Then, in section 4.2, it is discussed what exact measures are used for making comparisons.

4.1 Making the reported findings comparable

4.1.1 Procedure and resulting material

Figure 4 illustrates the process of making the reported findings comparable. Starting points were the reports with problem descriptions as they were handed in by the teams (figure 4, blocks at the top). The reports were used and compared without any further clarification and explanations by the teams. The comparer (the first author of the present article, who had not been involved in conducting or analyzing any of the test sessions) read through all reports and (as much as possible) annotated each moment of difficulty reported in a problem description with a unique identification number (id). However, this proved not always to be possible. In some cases, problem descriptions referred to *n* unspecified users having experienced the problem, or

What's the problem?

Studies on identifying usability problems in user tests

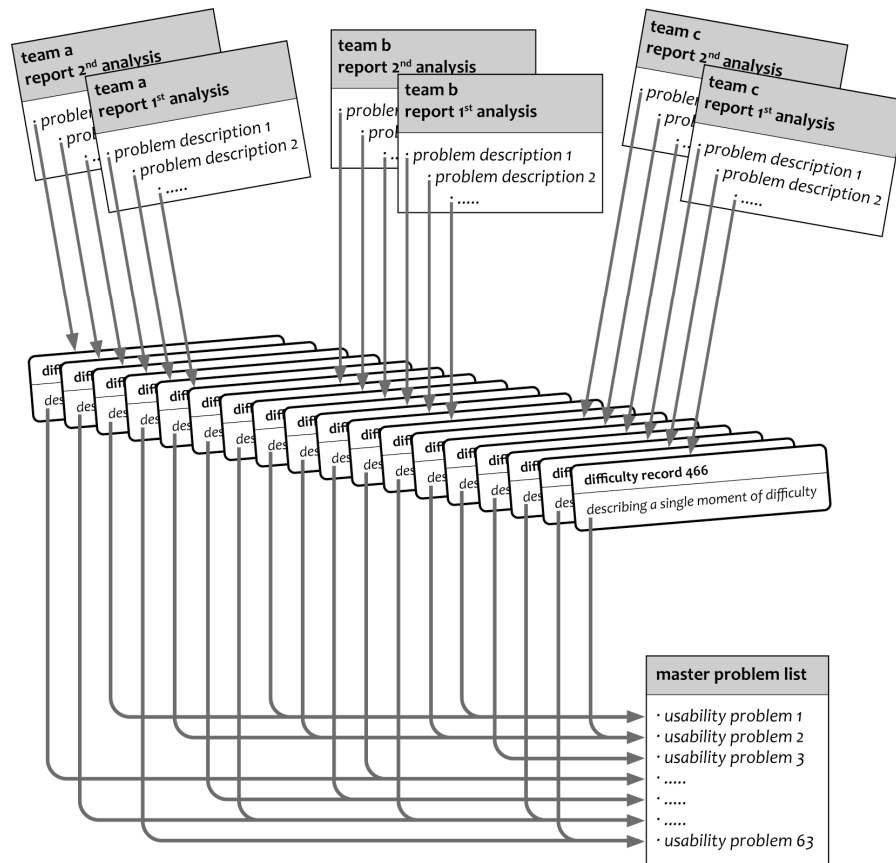


Figure 4 Problem descriptions in team reports (blocks at the top) were re-formatted into difficulty records (blocks in the middle). Then similar moments of difficulty were grouped in order to arrive at a master list of usability problems (block at the bottom).

contained no reference to any specified user or specified number of users at all. Figure 5 specifies the applied decision procedure for assigning ids to moments of difficulty. After ids were linked to the specific problem descriptions, a difficulty record was created for each id (figure 4, blocks in the middle). Figure 6 provides an overview of attributes of problem descriptions that were used as *fields* in the difficulty records. This part of the process resulted in a total of 466 difficulty records.

After entering as many attributes as possible for each of the moments of difficulty, usability problems were defined based on similarity. This was done in an iterative, data-driven process where the actual problem definitions emerged from the descriptions of the moments of difficulty themselves. Figure 7 shows some example usability problems.

Moments of difficulty had to be similar in fairly high levels of detail for considering them to be the same problem. For example, one might argue that the first three problems mentioned in figure 7 essentially are instances of one underlying problem, namely a problem with the rotary knob. However, important in many comparative user studies and in studies conducted in design contexts are inferences about causes of reported problems. In this example case, the three mentioned problems led to different explanations of why the rotary-knob related problems occurred; in case of the first problem, participants *in their attempts to stop the cooking process* tried to set a very low temperature by turning the rotary, whereas in the second case participants most likely assumed that the cooking process could *directly be stopped, by pressing the rotary knob*. These problems refer to two different strategies for trying to stop the cooking process. As to the second and third problem: they both start from the same *wrong use of the rotary knob (trying to press it, whilst this was not possible)*.

if a problem description explicitly referred to a moment in only one specified user's interaction (i.e., one moment of difficulty) this was recorded as one single id (and thus as one single difficulty record), with a reference to that specific user;

if a problem description referred to moments in all users' interactions, these were recorded as individual difficulty records for each individual user (so in case of a difficulty for all 8 users, this turned into 8 difficulty records);

if a problem description referred to moments in n specified users' interactions (where $n <$ the total number of users), the reported difficulties were recorded as n difficulty records each referring to an individual user;

if a problem description referred to moments in n unspecified users' interactions (where $n <$ the total number of users), it was recorded as one single difficulty record, and a reference was made to 'n users', without specifying the users (because they were unknown).

if a problem description referred to moments in a not specified number of unspecified users' interactions, it was recorded as one single difficulty record, stating 'unspecified' in the database field referring to users.

Figure 5 Checklist that was used for deciding how to create difficulty records from problem descriptions.

In case of the second problem this was done with the aim of trying to *immediately stop the cooking process*, whereas in case of the third problem, this was done to *select a menu item or to make a menu setting*. Thus, these three problems are likely to have different causes and it is very likely that in a design context these problems would lead to

What's the problem?

Studies on identifying usability problems in user tests

different interface revisions. Identifying these three problems as one single problem might lead to false impressions of agreement similar to those reported by Hertzum, Jacobsen and Molich (2002). The process described above resulted in a list of 63 different usability problems (figure 4, block at the bottom).

Attributes (fields)	Possible values
1. ID	Unique number.
2. Team that reported the problem	<A>, , <C>
3. Participant	<Name> and <participant number>, or <unspecified>, or <number of participants>
4. Data analysis approach	<Lab specific> or <SlimDEVAN>
5. Task	<Task number> and short phrase indicating the task (e.g., <Stop the cooking process>)
6. Description of difficulties and causes assumed by the evaluator	Copied from the teams' reports in unedited form
7. The action stage at which the problem occurred	Action stages based on Hartson (2003): <Planning>, <Translation>, <Physical>, <Outcome>, <Assessment>, <Independent>
8. The context in which the problem occurred.	Exact reference in grammar-form to a screen image on the product: Cooking_mode_setting (Tab: Cooking, Menu: menu where mode for cooking can be set)
9. Target context (i.e., screen image that would have been shown had the participant performed the correct action in the context of the provided task)	Exact reference in grammar-form to a screen image on the product: Cooking_mode_setting (Tab: Cooking, Menu: menu where mode for cooking can be set)
10. Action that would have been correct in the context of the task and would have led to the target context.	Specified in loosely defined grammar that indicates the required steps in detail. For example <select_cooking_OK> (i.e., participant has to use the rotary labeled 'select' to select the Cooking tab and then press the button 'OK' to confirm the selection).
11. Action that was performed by the participant	Specified in loosely defined grammar that indicates the steps performed by the participant in detail. For example <press_select> (i.e., participant pressed the rotary labeled 'select').
12. Relevance to the problem of a number of high level issues (variables).	For each variable, relevance: <Yes>, <No>, <Maybe>. Multiple variables can be relevant, at least one is relevant. Variables were: <ul style="list-style-type: none">• manually setting some cooking parameters• making settings for selecting recipes• actions for naming self-defined recipes• actions for starting a cooking process, and for editing or stopping an ongoing cooking process• issues related to making general settings, like the clock time etc.• issues related to defrost settings• issues related to low-level issues on how to use buttons and menus• issues related to the meaning and use of the 'Back' button

Figure 6 Attributes of moments of difficulty as specified in the fields of the difficulty records.

As the choice of the level of abstraction of problems is to some extent arbitrary and can be expected to influence agreement levels, a second way of categorizing problems was used as well. This other way of categorizing was purely based on the higher level issues as specified in the usability problem attributes (see figure 6, attribute 12). In this higher level categorization, problems were characterized by combining (mostly pairs of) higher level issues that were marked as relevant to the reported moment of difficulty. For example, if a moment of difficulty related to the issues *manually setting the cooking parameters* (abbreviated as *Cooking*) and *low-level issues on how to use buttons and menu* (in short *Interaction techniques*), it would be categorized as <Cooking>-<Interaction Techniques>. In cases where only one issue related to the moment of difficulty it was characterized as, for example <Cooking>-<Only>. In rare cases of more than two issues, all possible combinations of two marked issues were treated as separate categories (this was done to get all data in the same shape). This resulted in a total of 35 high-level problem category pairs. For most analyses the (detailed) 63 problems were used. The 35 high-level problem category pairs were only used for comparing agreement measures.

4.1.2 Findings on consistency and inspectability

The process of creating a master list of usability problems proved to be hindered by how problem descriptions were structured (inconsistent formats), by the formulations that were used, as well as by the reports' lack of inspectability.

Uses rotary to stop the cooking process
 Presses rotary knob to stop the cooking process
 Presses rotary knob to select a menu item or set time
 Participant hesitates to select EDIT for prolonging the cooking time.
 Inefficient having to do so many actions for making settings: composing a name
 Setting wintertime should not be done by just changing the hours and minutes
 Inconsistency between menu options with and without default values
 Misunderstanding that BACK in tabs menu displays main menu showing the clock time.
 Participant needs to be able to make longer names for own recipes.

Figure 7 Examples of usability problems (from the master list of usability problems).

Inconsistencies in problem formulations Sometimes problem descriptions were formulated in behavioral terms of difficulties encountered by users (e.g., Figure A4, Appendix A: "... users continued interacting thinking that the oven was still cooking" or "... one user tried to reduce the temperature... "). In other cases problem descriptions were formulated in terms of problematic product features; then it was sometimes unclear whether any of the users actually experienced the problem or what exact difficulty they had encountered (e.g., Figure A4, Appendix A: "Left hand arrow indicating that there is a submenu available is not clear"). In yet other cases, behavioral descriptions of problematic interaction episodes were given, but the difficulties themselves were not

What's the problem?

Studies on identifying usability problems in user tests

described separately. This sometimes made it difficult to infer what exactly the researchers thought the difficulty was or whether they thought that multiple moments of difficulty had occurred (e.g., Figure A1, Appendix A): "... Pressed rotary knob, turned. Presses 'stop' but puzzled when she saw... (etc.)".

The *inconsistencies in problem descriptions* hindered the construction of a master usability problem list. In cases where one of a team's report described a problem in behavioral terms, and the other described it in terms of a problematic product feature, it often proved to be difficult to decide whether the descriptions actually referred to the same moment of difficulty; to be able to do so a comparer² has to infer causal relationships between observed behavior and problematic product features. Similar problems occurred in cases where problematic interaction episodes were described without separate descriptions or marking of difficulties. In order to find out to what extent such a description refers to the same moment of difficulty as other problem descriptions formulated in terms of an encountered difficulty, a comparer has to infer what the analyst may have concluded about how many (and how many types of) difficulties are embedded in the described interaction.

Useful information complementary to difficulty descriptions The core of the problem descriptions in the teams' reports (i.e., descriptions of difficulties and of problematic features) was often provided with complementary information. Such additional information sometimes proved to be essential (and often at least very helpful) for a better understanding of the problem. For example, *mentioning the task* in which a difficulty occurred provides context that can help in envisioning the situation in which the difficulty occurred. *Suggestions for interface improvements* or *inferences about how design elements may have caused the problems* may implicitly detail difficulty description (e.g., Figure B2, Appendix B: the description of the difficulty only states "... User presses rotary knob to stop the oven", whereas the inference about what may have caused the difficulty includes the statement: "... The rotary knob is the most dominant element among the controls, so that the user is directed to that without much intention." By stating this, the researcher implicitly details the difficulty description by suggesting that the user's focus of attention may have been at the wrong place.)

Thus, information complementary to the core of the problem description can be useful to a better understanding of the observed interaction. However, in a number of cases another problem then showed up. For example, in some cases *suggestions for solutions* or *inferences about possible causes* were not linked to specific observed difficulties on a one-to-one basis; instead, a group of inferences about causes were linked to a group of difficulties or reasons of difficulties. For example, in Figure A3 (Appendix A), it is unclear whether the researcher intended to relate suggestion 3 about users' preferences for a stop/start button to one of three mentioned reasons or only to the main problem ("... Users find the stop-button easily, press the button, but are then confused by the

² From now on the person who makes the comparisons between the team reports will be referred to as the 'comparer' (for reasons of brevity).

feedback the oven provides").

Raw descriptions of what users said during or after interactions also provided complementary information that helped in better understanding ambiguous problem descriptions. For example in Figure B4 (Appendix B): the main problem was formulated as "... When alarm is set it is not clear if time indicates time until alarm or indicates the actual time." The comment the user gives in the second finding at the bottom of the page "... I was not sure if duration was the total time or the time left" provides extra information that can be taken into account in trying to interpret the (more or less cryptic, main) problem formulation.

Inspectability of data analyses In some team reports, some of the raw descriptions referred to above were included and clarified reported problems. However, in many cases raw descriptions were missing and could only be found in the log overviews representing observed interactions. In order for that to be of any use, it should be possible to exactly trace back which raw descriptions relate to which problem. This relates to the issue of inspectability. Inspectability of data analyses can sometimes alleviate the problems mentioned above. For example, if the description of a problematic feature has some kind of reference to a specific moment that is captured in some representation of an interaction, that interaction may be re-inspected to find out what observation lies at the basis of the problem.

The primary reference needed for inspectability always is a *reference to the specific user* that encountered the difficulty. In cases where it is not specified which user encountered a difficulty, it becomes a very tedious and difficult job to go through all interactions and try and identify the exact session and moment at which the difficulty occurred. In those cases where this was tried, it usually ended up with various candidate moments from multiple users' interactions. No further information was then available for a better understanding of problem descriptions.

In addition to references to users, *references to tasks* also proved to be very helpful. Not only because the task description in itself helps in envisioning the context in which a difficulty might have occurred (see before), but also because it makes it possible to search in a more focused way for the specific interaction in which a difficulty has (or might have) occurred. In many cases such a reference to a task is not needed, because it is almost obvious in which task a difficulty must have occurred (e.g., Figure A4 (Appendix A): it is very likely – though not certain – that the difficulty "... after the user presses Stop, there is no feedback that the oven has stopped cooking" refers to the task in which the user is asked to stop the cooking process). However, in some cases this is less clear (e.g., Figure A4 (Appendix A): the problem *left hand arrow indicating that there is a submenu available is not clear* could have occurred in many tasks). In general, information about the task in which a difficulty occurred helped in finding back interaction episodes when overviews of interactions were available. Again, this was helpful because in such overviews complementary information could be found that helped in understanding the problem (e.g., verbal utterances of users, or for example, sequences

What's the problem?

Studies on identifying usability problems in user tests

of actions that helped re-constructing what the state of the product must have been at the time when the problem occurred).

4.2 The comparisons

The previous section described how problem descriptions were made comparable. This was done by first re-formatting them into uniformly structured difficulty records and by then constructing a master list of usability problems. This process of making findings comparable allowed for the comparisons that are described in the following section. First, comparisons between (findings from) the teams' initial and reference analyses will be discussed. Quantitative comparisons are made on the number of problems identified in each of the analyses, and the amount of overlap in problems was determined. An analysis is given on why certain problems were reported in one analysis and not in the other. This provides some information on consistency of teams, when re-analyzing interactions. Experiences in trying to trace back causes of inconsistencies are then discussed and shed a light on inspectability of reports.

Next, comparisons between teams are discussed, for the initial analyses as well as for the reference analyses. Quantitative comparisons are made of what was specified about each difficulty in the teams' problem descriptions, as well as about the extent to which teams reported similar or different problems (agreement or consistency across teams).

4.2.1 Comparing problems reported in a team's subsequent analyses

For comparing how many problems the teams' subsequent analyses produced, a measure of *thoroughness* (Hartson, Andre and Williges, 2001) was used, and *overlap* in identified problems was examined (i.e., the number of problems that a team found in both analyses, divided by the total number of problems they found in the two analyses). For calculating thoroughness and overlap, usability problems (from the master list of usability problems) were used as units of comparison. In addition, all problems that were uniquely identified either by a team's initial analysis or by its reference analysis were further inspected to trace back reasons of uniqueness. For that inspection, usability problems were traced back to the difficulty records on which they were based and if necessary to the teams' original problem descriptions in their reports.

Thoroughness. Hartson, Andre and Williges (2001) define thoroughness as *the number of real problems found divided by the number of real problems that exist*. In this case, it was assumed that all problems identified are real, as we had no reference criterion to determine whether problems are real or not. The *number of real problems that exist*, is defined here as the sum of all problems found by all three teams ($\text{teamA} \cup \text{teamB} \cup \text{teamC}$), using both their initial and reference analyses ($\text{teamX}_{\text{lab-specific}} \cup \text{teamX}_{\text{SlimDEVAN}}$). Figure 8 presents the results of the calculations.

	Initial analyses (lab-specific)	Reference analyses (SlimDEVAN)
$A/(A \cup B \cup C)_{\text{lab-specific} \cup \text{SlimDEVAN}}$	26/63 = 41,3 %	29/63 = 46,0 %
$B/(A \cup B \cup C)_{\text{lab-specific} \cup \text{SlimDEVAN}}$	23/63 = 36,5 %*	27/63 = 42,9 %
$C/(A \cup B \cup C)_{\text{lab-specific} \cup \text{SlimDEVAN}}$	23/63 = 36,5 %	33/63 = 52,4 %

* Because for the reference analysis with SlimDEVAN the results of only 6 (instead of 8) participants were reported, the measures for the team's initial report are based on the results of the same 6 participants.

Figure 8 Thoroughness of data analyses (usability problems as unit of comparison)

For all teams, the second (reference) reports describe a larger number of problems than the initial team reports. Thus the second analysis must have revealed problems that were not revealed in the initial analysis. However, based on the summative figures of thoroughness it cannot be excluded that the initial analysis also identified some unique problems. Thus, as a next step, which analyses yielded unique problems, and to what extent, will be examined.

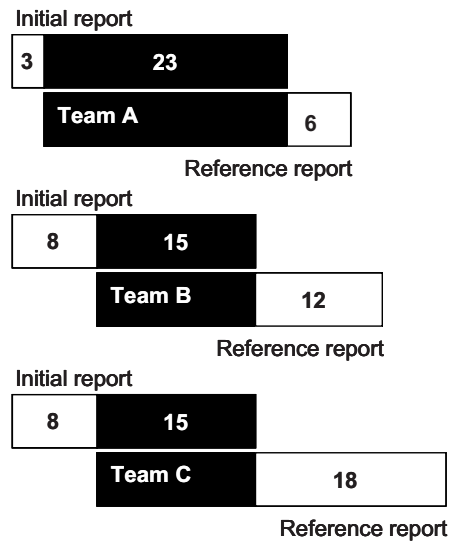


Figure 9 Number of problems identified in the teams' initial reports and in the teams' reference (second) reports. The black areas indicate overlap in problems between the first and second reports. Because team B's SlimDEVAN report reported results of only six (instead of eight) participants, the measures for the lab specific approach are based on the results of the same six participants.

What's the problem?

Studies on identifying usability problems in user tests

Overlap and uniqueness Figure 9 shows that for all three teams, both the initial analyses as well as the reference analyses identified unique problems. Because the thoroughness of reference analyses was always higher than the thoroughness of initial analyses the number of unique problems was always higher for the reference analyses. Within-team consistencies (i.e., the proportion of overlap depicted in figure 9) varied considerably across the three teams: 72% ($=23/(23+3+6)$) for team A, 43% ($=15/(15+8+12)$) for team B and 37% ($=15/(15+8+18)$) for team C.

Inspecting why problems were reported in only one of the teams' analysis reports

Uniqueness of problems extracted in only one of the analyses can be explained in at least two ways. It can be due to methodological differences in the subsequent analyses (e.g., the focus of a data analysis procedure or the way usability problems are defined), or it can be due to something like 'chance' (because researchers, as part of the measurement instrument, can not be expected to be equally concentrated during all situations in their analysis). Below, unique problems are studied in more detail by trying to find out to what extent the specific moments of difficulty on which they were based had been observed and analyzed in both analyses, and how these were further interpreted in each of the analyses. For that, problems were first traced back to the difficulty records on which they were based and from there, if necessary and possible to available interaction overviews that provided detailed insight into what actually happened. For example, if a problem was identified in the reference analysis, but not in the initial analysis, one could search in the reports of the initial analysis to find out whether the moment of difficulty had been observed at all and had been interpreted differently, or whether it seemed to have been overlooked. Thus, if a moment of difficulty was uniquely found in the reference analysis and it was not possible to find anything back about it in the reports of the initial analysis, this tells something about the (un) inspectability of the report from the initial analysis; after all both analyses were based on the same observed interactions.

For 57 (out of 102) moments of difficulty that were uniquely based on one of the two reports it appeared to be impossible for the comparer to find anything about it in the other report because of inspectability problems. In the remaining 45 cases there were no inspectability problems and reasons of uniqueness could be analyzed. On closer inspection, in 10 out of these 45 cases a problem had incorrectly been considered unique; in these cases the comparer had problems in interpreting the problem report which then led to difficulties in recognizing the usability problem underlying the moment of difficulty. In 35 cases uniqueness could be traced back to specific parts of the data analysis process. Below these categories are dealt with in more detail.

No inspection possible: unknown reasons of uniqueness Figure 10 shows how the 57 cases in which moments of difficulty appeared not be inspectable were distributed across the teams. There were two main reasons for hindering inspectability: (1) although the *problems* were described, the *moments of difficulty* on which they were

based were not described; in such cases one has no starting points for inspecting the other report, or (2) it was not possible to link back from a *problem description in the report* to a *moment of difficulty* reported in the other report.

Figure 10 shows that 45 (out of the 57) moments of difficulty with un-inspectable reason of uniqueness came from the reference reports (the sum of the values of the black bars), whereas 12 came from the initial reports (sum of white bar values). By analyzing the reasons of un-inspectability, it becomes clear to what extent these lie in the reference reports or in the initial reports.

Team A. In case of team A for five difficulties uniquely reported in the initial report it was not possible to find anything back in the reference reports. As *no complete interaction overviews* were available in the initial report (see Figure A1, Appendix A) the only way to find the moment of difficulty is, would be to re-view the video tapes. Therefore, essentially this is a problem of inspectability of the initial reports that did not guide enough in where to search for the unique difficulty in the other analysis. In case of six moments of difficulty uniquely reported in the reference report, the reason of uniqueness was not traceable, again for the same reason: in the initial reports there were no complete interaction overviews for verifying whether the interaction that lies at the basis of the difficulty had even been observed or not.

Team B. In case of team B for 15 moments of difficulty uniquely identified in the reference analysis, there was a lack of inspectability of the initial report. This lack of inspectability was a result of the fact that in the initial analyses (or at least in the initial report; see Figure A3, Appendix A) there were *no (relatively complete) overviews that represented the users' interactions* (there were only some example interactions embedded in the problem descriptions). Trying to inspect uniqueness of moments of difficulty would then imply re-viewing the video-recorded interactions for the specific task performance of the specific user again.

Team C. In case of team C, of 24 moments of difficulty were uniquely identified in the reference analysis, for which there was an inspectability problem in relation to the initial report (see Figure A4, Appendix A). A major reason for that was that in the initial report those moments of difficulty had *no reference to specific users*, some also had *no reference to a specific task* and *no interaction overviews* were available. Therefore, it was practically impossible to reliably trace back the moments of difficulty identified in the reference analysis to something in the initial reports if the same problem descriptions were not explicitly mentioned there. Also the cases of the three unique moments of difficulty that were identified in the initial reports only were caused by the lack of inspectability of the initial reports. These concerned *suggestions for solutions for which it was unclear whether any specific observed difficulty had been at the basis of it*. In four other cases, the comparer had no idea why the problems had not been recorded in the reference analysis: a *lack of interaction overviews in the initial reports* made it impossible to verify whether the problem had actually occurred or not (unless the video recordings would be re-viewed).

Thus, in summary, the initial reports proved to be less inspectable than the reference reports. Problems of a lack of inspectability of the initial reports occurred largely be-

What's the problem?

Studies on identifying usability problems in user tests

cause: initial reports provided no complete interaction overviews (team A, B and C), no reference to specific tasks (team C), no reference to specific users (team C) and because problems formulated in the initial report were written as suggestion with no apparent moment of difficulty mentioned (because of which it was unclear where to search in the reference reports or in available session logs of the initial reports, and the video would have to be re-viewed to see whether the difficulty actually occurred or not).

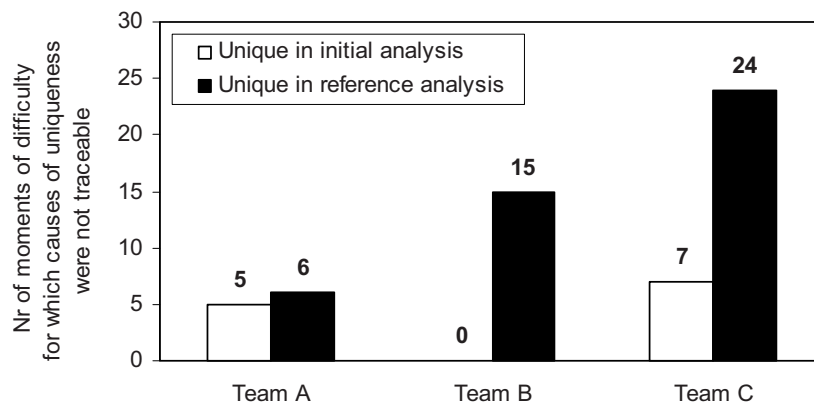


Figure 10 Moments of difficulty for which causes of uniqueness were not traceable.

On closer inspection not unique: 'comparer' problems. For ten (out of 102) moments of difficulty, the reason of uniqueness was caused by the fact that the comparer had had problems interpreting the problem report. In these cases, the inspectability of the reports made it possible to find out that the 'unique' moments of difficulty were not really unique. In eight of those cases the comparer had (on hindsight) made a wrong decision on whether a statement in a report should be interpreted as a problem or just as part of an overview of an interaction episode; this was the case for six records that were uniquely found in the reference analysis of team A and for two unique moments of difficulty in the reference analysis of team B. Here, problem descriptions were embedded in incomplete overviews of interaction episodes and were not separately and explicitly mentioned as such. In two other cases the comparer had interpreted a problem description in one analysis in a different way than he did in the other analysis. This happened with only one unique moment of difficulty from team B's initial analysis and with one unique moment of difficulty from team C's reference analysis. In case of team B this was due to an ambiguous problem formulation in the initial report, which was wrongly interpreted by the comparer.

Thus, to sum up, in some cases the comparer had made errors in interpreting problem descriptions. These errors related largely to ambiguous problem formulations (team B) and lack of explicit distinctions between logs and problem descriptions (team A and B) in the initial reports. This could only surface because in these cases the problem reports proved to be inspectable enough.

Unique problems: tracing back reasons of uniqueness. In the analysis of the remaining 35 (out of 102) cases for which the comparer had concluded that a moment of difficulty was uniquely reported and for which the reasons were traceable, the following categories of inferred reasons for differences emerged:

- 1 *false-positives*: moments of difficulty were reported but should not have been reported, because from the detailed analysis it has become clear that it is extremely unlikely that the problem actually occurred (and no other proof of existence was found other than the final problem description in the team's report);
- 2 *slips in the analysis leading to missed moments of difficulty*: the interaction in which the difficulty arose was observed in both analyses, but (for unknown reasons) was not recorded as a difficulty in one of the analyses even though it unambiguously represented a moment of difficulty;
- 3 *slips in the analysis leading to different problem descriptions*: missed observations or utterances leading to different interpretations of moments of difficulty and thus to different interpretations of what exactly the problems were;
- 4 *threshold differences*: differences in an analyst's decision of how difficult a difficulty should be for recording it as a moment of difficulty in the report;
- 5 *definition of 'usability problems'*: differences in what should lay at the basis of a usability problem (e.g., defining the fact that the user needed a hint as a moment of difficulty or not, distinction between problems of usability or other problems like software bugs).

Figure 11 shows that of the 35 unique moments of difficulty that were inspectable and regarded as being veritably unique, 23 were in the initial reports (sum of all values in the graph at the top of the figure) and 12 were in the reference reports (sum of all values in the graph at the bottom of the figure).

Team A. In case of team A (figure 11: grey bar), for the one moment of difficulty uniquely reported in the initial report that had been inspectable the reason of uniqueness was of type *threshold differences*. Based on the low number of inspectable moments of difficulty here, it may seem as if the reports of team A had been relatively uninspectable. However, one should bear in mind that team A had the smallest number of unique moments of difficulty, among which a relatively large amount of comparer problems (which is a problem not of inspectability but of problem descriptions in the initial report).

Team B. In case of team B (figure 11: white bars), most 'real' unique moments of difficulty for which the reason had been inspectable were in the initial report (top graph of figure 11). Reasons of uniqueness were mainly of types: *slips causing differences*, *threshold differences* and *definition of usability problems*. In addition there were a few of types *false-positives* and *slips causing misses*. In case of the *slips* uniqueness usually had to do with not hearing or not noticing user's verbal utterances containing clues that helped in interpreting the specific moment of difficulty.

Team C. In case of team C (figure 11: black bars), reasons of uniqueness were found to be in the categories *threshold differences* and *definition of usability problems*. In addi-

What's the problem?

Studies on identifying usability problems in user tests

tion, for eight moments of difficulty the reason of uniqueness was placed in the category *false positives* (seven of these were from the reference report).

Although the number of seven seems impressively large, it should be mentioned that these were seven instances of one single usability problem, and each of the instances was from a different user.

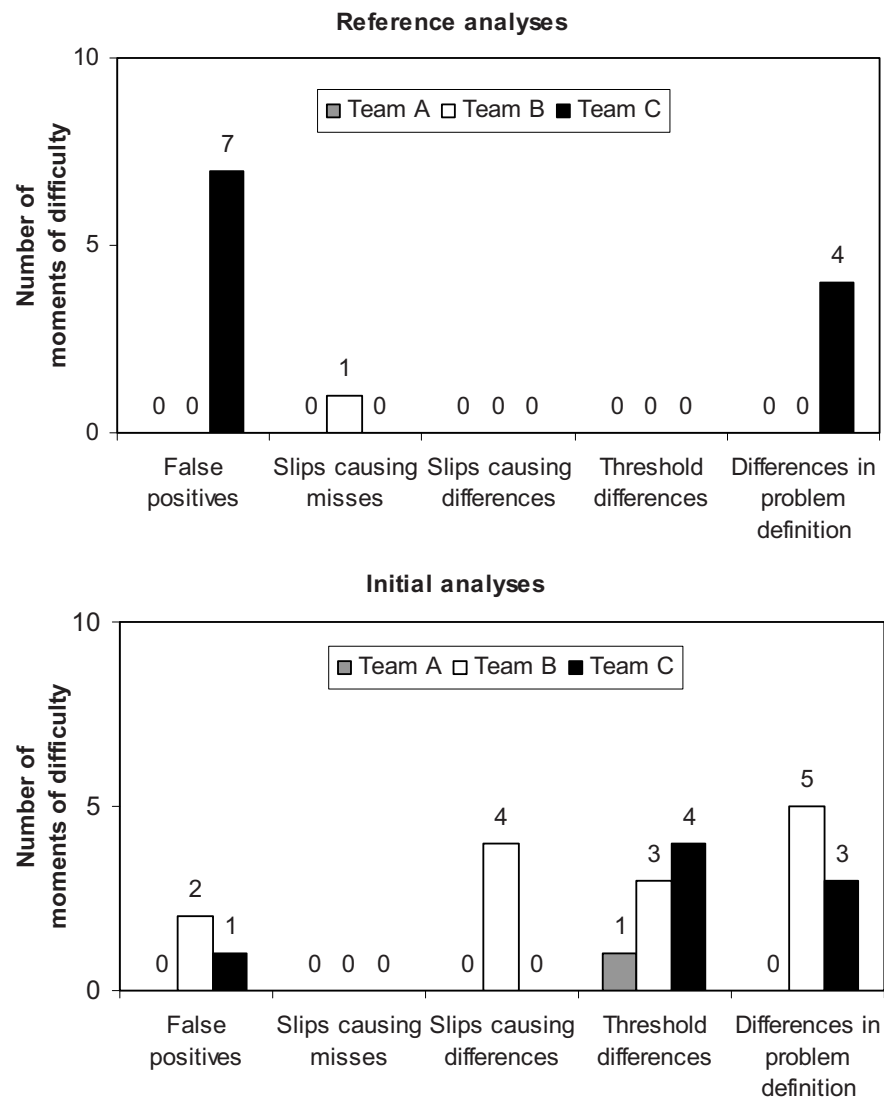


Figure 11 Analysis of the reasons of uniqueness of moments of difficulty.

In summary, inspectability of the reports allowed for tracing back causes of uniqueness in these 35 cases. Reasons for uniqueness included: *slips causing misses* (team B reference analysis), *slips causing differences* (due to for example missing user utterances) (team B initial analyses), *threshold differences* (team A, B, C), differences in *definitions of usability problems* (team B and C) and *false positives* in the reference analysis (team C).

4.2.2 Comparing the reported problems across the three teams

Some of the categories of reasons for uniqueness mentioned in the previous section seem to relate to issues of analysts working inaccurately (e.g., lack of attention), whereas others might be related to issues of differences in point of view, opinions or focus. For example, a more explicit and detailed procedure (as in the reference analysis with its checklist of signals of difficulty), may (but not necessarily does) bring about an implicit focus on specific elements of interactions or specific interpretations in formulating problems. If this would be the case, than one would expect to find less consistency between teams in the initial analysis (lab-specific analysis) than in the second (reference) analysis. Thus shifts in levels of agreement between teams when moving from the initial analyses to the subsequent analyses can tell something about the quality, in terms of biasing the results from the reference analysis. Below, levels of agreement across teams are calculated. First, comparisons are made of *which* problems were reported. Subsequently, comparisons are made on the *type of information that is given about* each problem.

Comparing which problems were reported For determining consistency in what problems each of the three teams reported, evaluator agreement measures are used. Hertzum and Jacobsen (2001) suggest the use of the any-two agreement measure in cases where the number of evaluators is relatively small. In this article, Hertzum and Jacobsen's (2001) definition is used, substituting evaluators by teams:

$$\text{the average of } \frac{P_i \cap P_j}{P_i \cup P_j} \text{ (over all } \frac{1}{2}n(n-1) \text{ pairs of teams).} \quad (1)$$

Similarly, agreement between x teams can be defined as:

$$\frac{P_i \cap P_j \cap \dots \cap P_x}{P_i \cup P_j \cup \dots \cup P_x}. \quad (2)$$

In equations 1 and 2, P_i , P_j and P_x are the sets of problems (or high-level categories) detected by team i , team j , and team x , and n is the number of teams.

Figures 12 and 13 show the any-two agreement measures for usability problems and for high-level categories respectively.

Due to the higher level of abstraction (and hence the smaller number of categories) the higher any-two agreement for high level problem categories comes not unexpected.

What's the problem?

Studies on identifying usability problems in user tests

On both levels of abstraction, any-two agreement in the initial analysis is about the same as in the reference analyses. If the largely prescriptive analysis and reporting techniques of the reference analysis would have introduced a bias in finding usability problems, a higher level of agreement would have been expected for the second (reference) analyses. No indications were found that the reference analysis leads to a specific bias in the teams' results. In other words, the level of agreement does not seem to be influenced by the fact that the teams' analysis methods used in the initial analyses were different, whereas those in the second analyses were similar.

Usability problems	Agreement between teams	
	Lab specific	Slim-DEVAN
Team A vs. B	11/38 = 28,9 % *	11/45 = 24,4 %
Team A vs. C	13/36 = 36,1 %	15/47 = 31,9 %
Team B vs. C	10/36 = 27,8 % *	16/44 = 36,4 %
Any-two agreement	30,9 %	30,9 %
Three agreement Teams A-B-C	(7/45 =) 15,6 %*	(9/56 =) 16,1 %

* Because for the reference analysis with SlimDEVAN the results of only 6 (instead of 8) participants were reported, the measures for the team's initial report are based on the results of the same 6 participants.

Figure 12 Agreement between teams (usability problems as unit of comparison)

High-level problem categories	Agreement between labs	
	Lab specific	Slim-DEVAN
Team A vs. B	12/23 = 52,2 %*	15/28 = 53,6 %
Team A vs. C	12/23 = 52,2 %	14/23 = 60,9 %
Team B vs. C	13/19 = 68,4 %*	17/31 = 54,8 %
Any-two agreement	57,6 %	56,4 %
Three agreement Teams A-B-C	(11/25 =) 44,0 %*	(13/31 =) 41,9 %

* Because for the reference analysis with SlimDEVAN the results of only 6 (instead of 8) participants were reported, the measures for the team's initial report are based on the results of the same 6 participants.

Figure 13 Agreement between teams (high level problem categories as unit of comparison)

Comparing 'what was specified about each problem' Figure 14 shows, for each of the analyses, an overview of the elements that problem descriptions consisted of in case of each of the three teams. For that, the fields *difficulty* and *cause* in each difficulty record (see figure 6, 6th attribute) were analyzed. The contents of these difficulty record fields had been taken literally from the teams' reports. These difficulty records were analyzed

to find out to what extent they mentioned (or otherwise referred to) the following problem description elements:

- 1 the *situation* in which the problem occurred (was the product status mentioned, was there a mentioning of preconditions for actions that were not met?)
- 2 the *user's observable behavior* at the time the difficulty occurred (what parameters was the user trying to set, what physical action was the user trying to perform?)
- 3 what the user *thought, felt* or *understood* (explicit mentioning of inferences about the user's reasoning, understanding, feelings or about what the user tried to achieve)
- 4 what the *effect of the difficulty* was (this relates to the effect of the problem on the product status, as well as effects on the user, like confusion, frustration, etc.)
- 5 inferences about what *product element* had *caused* the difficulty or should be redesigned to avoid it (explicit mentioning of product characteristics that are believed to have contributed to the difficulty, of the reason why they are believed to have contributed to it, as well as suggestions to change the functionality of the product).

Compared are the proportions of problem descriptions that contained statements referring to each of the mentioned descriptive elements. Thus, the figures are corrected for differences in absolute numbers between the initial analyses and the reference analyses.

Figure 14 shows that in the graph at the bottom, the relations between the bars per team (i.e., all bars of a certain color) are more similar than in the graph at the top. In other words, it seems that in the second analysis teams have produced more similar reports in terms of what they reported about problems.

Summarizing, the reference analysis procedure did not lead to more agreement on what usability problems were extracted, but it did lead to more consistency in what was reported about each of the problems. Teams reported more about the situation in which problems occurred and were more in agreement with each other with respect to the amount of problem descriptions containing behavioral observations and inferences about what users seemed to understand, feel and think. Especially team C, which originally reported relatively little about what users did, now reported more about that. The teams that in the initial reports mentioned only few possible causes of problems (teams A and B), now showed a considerable increase, whereas team C that already was at a high level, reported relatively less causes. Finally, for teams A and B, the reporting of effects of actions dropped to a very low level in the reference analysis, whereas team C stayed at an already relatively low level.

What's the problem?

Studies on identifying usability problems in user tests

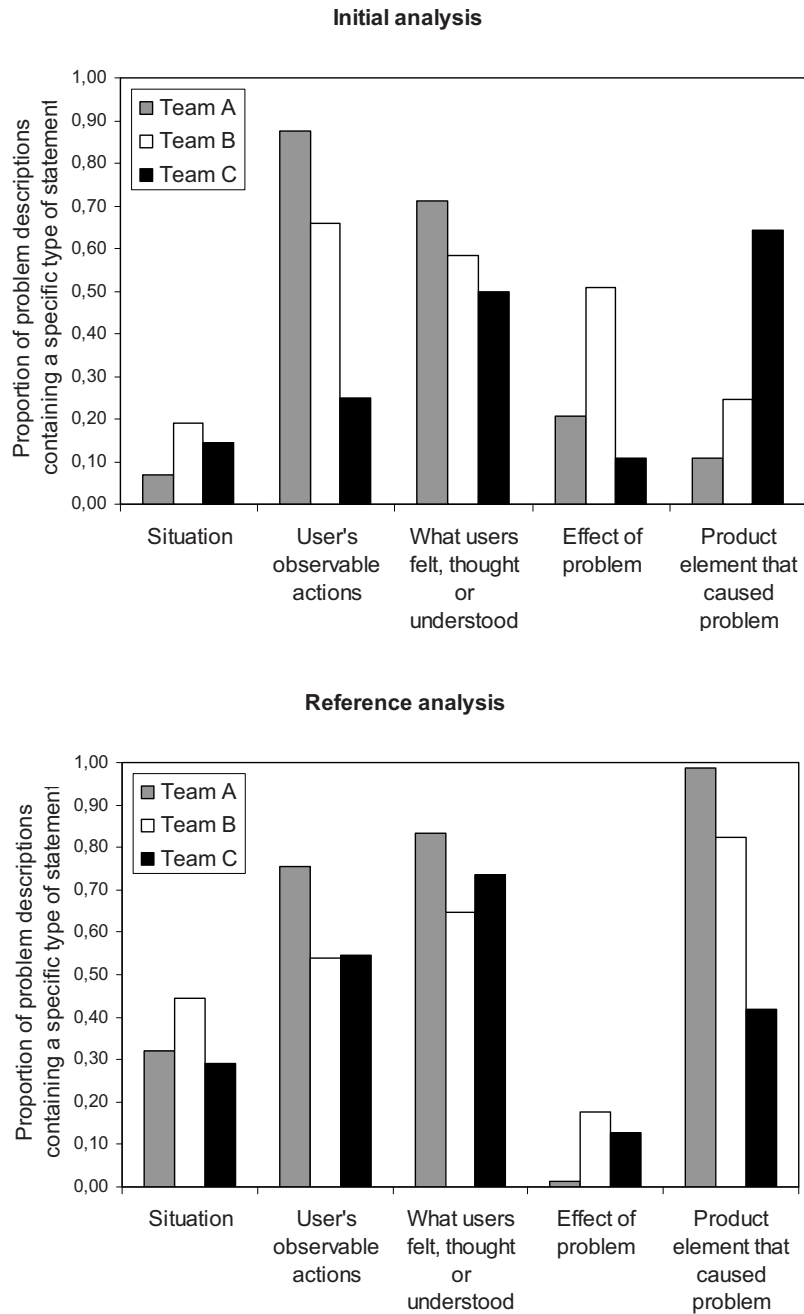


Figure 14 Overview of the descriptive elements that each of the teams used in their problem descriptions (proportion of problem descriptions that contained a specific statement).

5 Discussion and conclusion

The aim of the study was to investigate issues of consistency and inspectability of data analyses and reports based on usability tests meant to extract usability problems. Consistency of findings was studied for analyst teams who analyzed their observed interactions twice with time intervals of one to two months (the first time using their own usual method of analysis, the second time all using the same reference method of analysis). Inconsistencies were analyzed in detail in order to trace back their origin. Consistency across teams was studied in terms of levels of agreement on problems and high-level categories; it was determined to what extent these levels change when the teams, in their re-analysis, all use the same data analysis method. Also, it was explored to what extent the use of the same data analysis approach led to the use of similar descriptive elements in the teams' problem descriptions. By making the reported usability problems comparable as well as by analyzing reasons of inconsistencies, it became clear which problem report elements were important for (1) making reports inspectable and (2) making the reported findings traceable to the original observations. Below the findings on inspectability and consistency are discussed.

5.1 Inspectability

Inspectability of usability problem reports is important for laying bare those elements in a data analysis process that involve subjective analyst judgments. In this study it became clear that for comparative studies inspectability is also important for two other reasons. Firstly, it proved to be of crucial importance for understanding reported problems in enough detail to be able to decide on their similarity. Secondly, inspectability proved to be of major importance for tracing back reasons of why a team reported a problem in one analysis and not in the other. For example, problems of inspectability when trying to trace back the origin of uniqueness were present in 57 of the 102 problem records that were reported in just one of a team's analyses.

The absence of the following report characteristics proved to be a hindrance in tracing back findings to more primary data. In other words, to make usability problem reports more inspectable, the following issues can be helpful:

- inclusion of descriptions of a user's actual *behavior* in the description of a problem, e.g., descriptions of problematic user actions;
- making references to which users encountered a problem;
- making references to the task context or the (sub) goal that a user was trying to accomplish at the time of the problem;
- making clear distinctions between descriptions of problematic interactions and descriptions of interactions that are interpreted as proceeding without problems;
- inclusion of (or making references to) overviews of observed interactions (without substantial gaps), instead of only relying on problem descriptions;

What's the problem?

Studies on identifying usability problems in user tests

- inclusion of an account of a user's verbal utterances, as well as clear links between such account and interaction overviews or problem descriptions;
- inclusion of inferences about causes of problems (or suggestions for how to resolve problems) with clear links to behavioral problem descriptions.

5.2 Consistency

5.2.1 Across-teams consistency (subjectivity)

Comparison of what descriptive elements were used to report problems showed that the (SlimDEVAN) reference reports led to more across-teams consistency, than the initial reports. In other words, without using an agreed method for reporting problems, consistency across analysts on what elements are used to report problems may suffer; this in turn makes it more difficult to compare the usability problems identified by those analysts. In the present study, the SlimDEVAN method led to more consistency in the degree to which the teams provided descriptions of the *situations* in which problems occurred, of the *behavioral descriptions* of observable problematic actions, of accounts indicating what analysts *inferred users were feeling, understanding or thinking*, and in mentioning *product elements* that presumably had caused problems.

Although consistency in what descriptive elements were used to report problems was higher in the reference reports than in the initial reports, measures of consistency on *which* problems were identified tell a different story: any-two agreement levels were about the same for the initial and reference analyses. This implies that adopting similar data analysis approaches and using an agreed format for describing usability problems do not necessarily lead to more consistency in findings across teams. Any-two agreement of the teams (comparing usability problems) was 30,9% for both analyses. Based on our data, it is unclear whether the lack of agreement stems (1) from differences in the teams' original data (e.g., actual differences in interactions, because the teams used different test participants in different countries) or (2) from some team-specific (i.e., not method-specific) characteristic of the analysis (e.g., inherent subjectivity in the data analysis process).

5.2.2 Within-team consistency

For measuring within-team consistency two measures were used: (1) *thoroughness* of the first analysis and of the second analysis; (2) *overlap*: the number of problems that a team found in both analyses, divided by the total number of problems they found in the two analyses.

In the reference analysis (the second analysis) *thoroughness* proved to be consistently higher than in the initial analysis. Two possible reasons are: (1) problems identified in the initial analysis may have a higher chance of being identified again in the second analysis, while at the same time analysts have a second chance for identifying problems that were overlooked in the first analysis; (2) due to its characteristics the reference method is able to identify a larger number of problems. The present data do not allow determining to what extent these reasons contribute to the current findings.

The proportion of *overlap* of findings from the initial and subsequent analyses was found to be substantially different across teams. One can think of two possible reasons: (1) differences in levels of thoroughness between initial and subsequent analyses varied considerably (e.g., more *thoroughness* in the analyses increases the chance for more overlap); (2) the methods the teams used in their initial analyses differed in the extent to which they facilitate analysts to work in a consistent manner. Based on the available data it is unclear which of these reasons is most likely.

The reasons behind the findings on (within-team as well as across-teams) consistency all raised some questions. Due to issues of confounding in the set up of the study, the questions can not be answered just by using the data of the present study. In making within-team comparisons there was confounding due to the analysis methods the teams used as well due to an order effect (teams used different methods in their first and second analysis); in the across-teams comparisons, there was confounding due to the fact that the three teams worked from observations of different test participants. Data from a separate study described in Vermeeren, Koenderink-van Doorn and de Ridder (2006) suggest possible answers. These are dealt with below.

5.3 Follow-up study on causes of (in)consistency

In the study described in Vermeeren, Koenderink - van Doorn and de Ridder (2006) two pairs of students (in the context of a course on research methodology) were asked to analyze parts of the recordings from team B twice with an interval of about three weeks and to use SlimDEVAN for both their analyses. Students were free to decide how many and which sessions they analyzed. They were third-year (Bachelor) students of Industrial Design Engineering at Delft University of Technology and had no experience in formal user testing and data analysis. However, they had three years of experience in practical courses on user-centered design and evaluation of consumer products. Students were provided with the English language user manual of SlimDEVAN (Vermeeren, 2003), with an abbreviated Dutch language user manual as well as with published literature on DEVAN (Vermeeren, van Kesteren and Bekker, 2003; Vermeeren, 2004). In addition, the first author of the present article explained and discussed SlimDEVAN with each individual team of analysts. In addition, after they had performed their first session, student teams had their analysis checked by the first author of the present paper to identify misunderstandings of SlimDEVAN.

The student teams analyzed the sessions of their choice twice with a minimum interval of three weeks. During those weeks the students did not watch the recorded sessions nor did they review their analyses; largely they spent time on doing exams for other courses as well as on doing a literature search on the topic of comparing usability evaluation methods. Student team 1 decided to analyze the sessions of test participants 1, 2, 3, 4 and 5; student team 2 analyzed the sessions of test participants 2, 4, 5 and 6.

What's the problem?

Studies on identifying usability problems in user tests

The following measures were applied to study issues of consistency:

- across-teams consistency in the second analysis, in terms of agreement between findings of each student team and of lab team B (expressed as a proportion of the total number of problems identified by team B and the student team in their second analysis). Figures are based on the same sample of test participants for student team and team B). See figure 15 for the results.
- thoroughness of the students teams' first and second analyses in comparison to thoroughness of team B's analyses (again for the same sample of test participants). *Total number of usability problems that exist* is defined here as the total number of problems identified by all three lab teams and the two student teams in their first and second analyses (77 problems). See figure 16 for the results.
- within-team consistency in terms of overlap between first and second analysis in comparison to that of team B (for the same sample of participants). Overlap is expressed as the proportion of the total number of problems identified by a lab/student team in their two analyses. See figure 17 for the results.

In the discussion, the question was raised as to what had caused the limited agreement in the lab teams' findings: would it be due to differences in the observed interactions themselves (as the teams had used different participants) or to issues of analyst subjectivity. Even though each of the student teams and team B had analyzed exactly the same data (i.e., the same sessions of the same participants), the levels of agreement still are roughly in the same range as those of the lab teams (i.e., 24,4%, 31,9% and 36,4% for the SlimDEVAN analyses; see figure 12). This is contrary to the expectation one would have if the difference in original data would have been the most important factor for the lack of agreement. This suggests that analyst team-specific factors (e.g., subjectivity) play an important role in the lack of agreement. The fact that agreement levels between team B and the student teams were even lower than agreement levels between lab teams may be explained by the fact that these were based on data from a smaller number of participants.

Analyst combination	Participant sessions	Analyst agreement second analysis (SlimDEVAN)
Student team 1 vs. Lab team B	1, 2, 3, 4, 5	26,8%
Student team 2 vs. Lab team B	2, 4, 5, 6	29,4%

Figure 15 Analyst agreements of student teams and lab team B in the second analysis (proportion of the total number of problems identified by the lab team and the student

With respect to the observed increase of thoroughness from the first to the second analyses, the question arose: to what extent would this be due to the fact that the second analysis is a re-analysis or to some method-specific factor. In figure 16, the observed increase in thoroughness for the student teams (i.e., 1,3% and 3,9%) can only be due to the sole fact that the second analysis was a re-analysis. A similar increase in

thoroughness for lab team B would suggest the same underlying reason. However, figure 16 shows that the increase for lab team B is higher. This suggests that the increase in the labs' thoroughness is partly due to the fact that the analysis is a re-analysis (i.e., in the range of 1,3 - 3,9 %) and partly due to differences between the methods that were used in the first and second analysis (i.e., one of the methods being more thorough than the other). Further research is needed to substantiate this.

Analysts	Participant sessions	Thoroughness 1st analysis	Thoroughness 2nd analysis	Increase in thoroughness
Student team 1	1, 2, 3, 4, 5	31,2% (SlimDEVAN)	32,5% (SlimDEVAN)	1,3%
Lab team B	1, 2, 3, 4, 5	29,9% (Lab specific)	35,1% (SlimDEVAN)	5,2%
Student team 2	2, 4, 5, 6	49,4% (SlimDEVAN)	53,3% (SlimDEVAN)	3,9%
Lab team B	2, 4, 5, 6	26,0% (Lab specific)	32,5% (SlimDEVAN)	6,5%

Figure 16 Thoroughness of teams compared across 1st and 2nd analysis. Total number of usability problems that exist is 77.

Analysts	Participant sessions	1 st Analysis	2 nd Analysis	Within-team consistency
Student team 1	1, 2, 3, 4, 5	SlimDEVAN	SlimDEVAN	63,3%
Lab team B	1, 2, 3, 4, 5	Lab specific	SlimDEVAN	42,9%
Student team 2	2, 4, 5, 6	SlimDEVAN	SlimDEVAN	71,7%
Lab team B	2, 4, 5, 6	Lab specific	SlimDEVAN	36,4%

Figure 17 Within-team consistencies of student teams and lab team B compared (proportion of the total number of problems identified by a lab team/student team in their two analyses).

The question regarding within-team consistency in terms of overlap was whether this would relate to thoroughness or to differences in methods (as some methods may make it easier to work in a consistent manner than others). The findings in figures 16 and 17 suggest no clear relation between thoroughness and within-team consistency; the highest and lowest within-team consistency are found for the analyses with the highest and lowest thoroughness, but within-team consistency of student team 1 is also high without their thoroughness being quite high. Note that within-team consistency is high for both student teams and low for lab B. This raises questions about the role of the used analysis method in consistency across subsequent analyses: What role do the characteristics of the used analysis method play in this? Would a comparison of two iterations of any method give higher consistency than a comparison of a method with a

What's the problem?

Studies on identifying usability problems in user tests

different method? To what extent do characteristics of the analyst teams play a role? Further research is needed to answer these questions.

5.3.1 Conclusions

The findings in the case study together with those from the complementary study suggest that:

- the levels of agreement between analyst teams relate to analyst team-specific characteristics, rather than to characteristics of the used analysis methods or to differences in the original data. In other words there seems to be considerable inherent subjectivity in findings from a usability test based on identifying problems and consistency across teams is not very likely to be improved by using specific analysis methods;
- re-analysis of the same data is very likely to lead to a slightly higher thoroughness in the second analysis even if the second analysis is done one or two months later (in this case in the range of 1-4%);
- within-team consistency can vary considerably (in this case between 37% and 72%) and may depend on the methods used or on 'who does the analyses'.

5.3.2 Implications

In usability studies that are based on comparing problems, it is important that all data are analyzed by the same analysts/analyst teams. This is important because of the inherent subjectivity in extracting usability problems from observations. With appropriate methods that conform to the characteristics as present in SlimDEVAN and as mentioned in section 5.1 the findings can be made inspectable. Inspectability is important for laying bare subjective parts in data analyses and for making sure that outside reviewers can falsify or confirm the findings. In comparative usability studies inspectability is also important for a thorough understanding of the reported problems. This is crucial for being able to decide on similarity of problems.

chapter 4 **Identifying usability problems in product tests: effects of using cardboard and software prototypes**

abstract

Findings from usability tests of a functioning TV-video recorder combination were compared to those of a cardboard mockup and a VisualBasic computer prototype. Usability data were analyzed in a detailed and structured manner. Focusing on high-level task performance measures like the number of task failures and counts of usability problems only small differences were found between the findings from the product, prototype and mockup. However, in more detailed analyses several relevant differences in participants' behavior could be identified. In case of the cardboard mockup, the fact that the interaction was mediated by a facilitator not only improved the clarity of participants' verbal information, but also resulted in participants not showing typical behavior like repetitively pushing buttons not leading to any response. Other differences seemed to be related to specific properties of the prototypes in comparison to the product, like the visual prominence of feedback (in particular for the mockup), the limited functionality of both prototypes and the fact that, in case of the software prototype, both the remote control and the menu it controls were shown on the same computer screen.

This chapter is in review:

Vermeeren, A.P.O.S., de Ridder, H., van Doorn, A.J. (submitted). Identifying usability problems in product tests: effects of using cardboard and software prototypes. (submitted to Journal of Usability Studies, October 2008).

What's the problem?

Studies on identifying usability problems in user tests

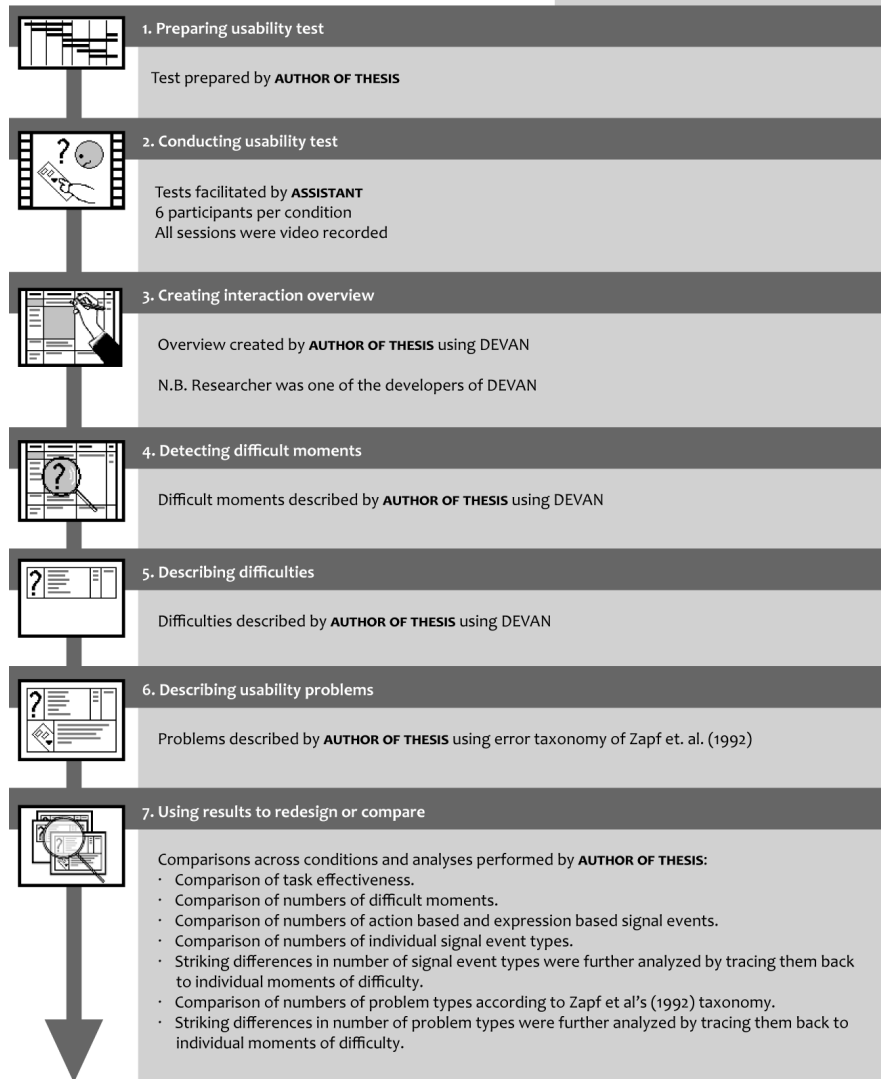
Overview of characteristics of the studies reported in chapter 4

Product

- Philips TV-video combination (TV-VCR)

Three conditions

- Functioning product
- Cardboard mockup
- Software prototype on computer



1 Introduction

A number of studies have been conducted in which usability problems identified in usability tests with low-fidelity prototypes are compared to those identified in tests with functioning products or high-fidelity prototypes (see Sauer, Franke and Ruettinger, (2008) for an overview). Generally, the aim of such studies was to study to what extent (for example in a product development project) low-fidelity prototypes can adequately be used to predict usability problems that will occur once the finished product would be used. In a recent publication on assessing usability evaluation methods (UEMs) Hornbæk (in press) discusses the problematic assumptions that lie at the basis of many of such comparisons. The most common assumption he mentions is that just focusing on summative measures like counting problems is sufficient. The limitation of such an approach is that...

“... different kinds of problems - for example with respect to generality, type, aspects of the interface covered, or clarity - are given equal weight when counted. This limitation becomes serious if different UEMs lead evaluators to produce problem descriptions of different kinds”.

This limitation may also apply to comparisons of usability problems from usability tests with different kinds of prototypes. Hornbæk argues for a more careful analysis of the contents of problems so that similarities and differences between problems can be determined (e.g., by using problem classifications or taxonomies). Furthermore, in the literature a number of publications have appeared that make clear that there are unavoidable subjective components both in extracting problems from observations and in comparing and/or matching usability problems (e.g., Lavery, Cockton and Atkinson, 1997; Cockton and Lavery, 1999; Hertzum and Jacobsen, 2001; Vermeeren, van Kesteren and Bekker, 2003; Howarth, Andre and Hartson, 2007; Hornbæk and Frøkjær, 2008a; Vermeeren et al., 2008). In most of the studies cited by Sauer, Franke and Ruettinger

What's the problem?

Studies on identifying usability problems in user tests

(2008), differences in the number of usability problems found through testing a functioning product (or high-fidelity prototype) and testing a (low-fidelity) prototype are reported to be small. However, these studies did not report on one or more of the following methodological issues: systematic procedures to uncover the nature of the differences in problems by looking at their contents; measures for improving consistency and dealing with subjectivity; procedures for extracting and matching usability problems.

The present article describes a study in which the findings from usability tests with a functioning product (a combined TV-video cassette recorder) are compared to those with a cardboard mockup as well as an interactive software prototype on a PC. The aim is to explore how characteristics of prototypes and the active role of the facilitator that is needed for many prototypes may influence the resulting list of problems. All recorded test sessions were processed by the same analyst using the DEVAN (DEtailed Video ANalysis, Vermeeren et al. 2002) method, which was especially developed for improving consistency and inspectability in extracting usability problems from video recorded usability test sessions (e.g., Vermeeren et al., 2002; Vermeeren et al., 2008). As the DEVAN analysis allows for comparisons at various levels of detail, sessions are first compared at the following high-level task performance measures: *task effectiveness* and number of so-called *difficult moments in interactions*. Subsequently, the findings were studied in more detail to gradually uncover similarities and differences that possibly relate to characteristics of the prototypes used. This was done in two different ways: 1) by using lower-level data provided in the course of the DEVAN analyses, and 2) by conducting a causal analysis on the difficult moments. DEVAN does not include causal analyses. Zapf et al.'s (1992) theory-based error taxonomy was used for this because of its ability to relate identified types of errors to characteristics of a user's behavior.

2 Methods

2.1 The product

The object of evaluation was a Philips combined TV-video cassette recorder (VCR) (see figure 1). The video recorder part is located above the TV screen, behind a lid that can be opened with the only visible button on the front. The user can control the product with a remote control (RC) and a small number of buttons behind the lid. The remote control has buttons on either side of it. The quick reference card that participants received refers to these sides as the 'daily side' and the 'special side'. Product feedback is given in the form of LED feedback information displayed just above the TV screen and as on-screen information (see figures 2 and 3).

The 'daily side' of the RC contains a rotary for switching TV channels. In addition it contains buttons for standby, mute, volume (plus and minus), invoking a 'timer for today'

menu, play, rewind, wind (on most VCRs this is called 'fast forward'), stop and record. The exact functioning of the rewind and wind buttons is different from that of most other VCRs; it depends on the speed and direction of the tape at the moment of using them. For example, pressing rewind while a tape is playing, results in a still image. Pressing it again results in reversed play and pressing it yet again results in rewinding with visible images. Pressing rewind when the tape is not moving (and there is no still image) results in rewinding at fast speed, without visible images. Note, that contrary to most VCRs, a still image is not evoked by pressing the pause/stop button, but by pressing the rewind button when playing a tape. The 'special side' of the remote control contains functions for teletext, various menus, cursor keys and an 'OK' button for using menus, numbered buttons, and additional functions like switching to an external source etc.

Three broadcasted channels (Netherlands 1, 2 and 3) were simulated by playing pre-recorded broadcasts on VCRs connected to the antenna input of the TV-video combination. As a consequence, all participants had the same broadcasted programs to choose from. The pre-recorded programs were selected to be representative for broadcasts that typically fitted the profiles of the Dutch channels at the time (i.e., Channel 1: religious programs; Channel 2: sports, quizzes, soap series; Channel 3: news, current affair programs, culture, etc). Teletext was not available.



Figure 1 The TV-video combination, showing the on-screen menu 'Timer for today' ('Timer voor vandaag'). On the right the two sides of the Remote Control: the 'daily' side (with rotary for 'switching between channels') and the 'special' side.

What's the problem?

Studies on identifying usability problems in user tests



Figure 2 Schematic view of the LEDs and labels above the screen.



Figure 3 The timer menu. Listed items are: Timer block, Channel, VPT <no>, Start time, Stop time, Date, VPS <no>, Timer for <recording>, Timer <off>.

2.2 The prototypes

Two prototypes were especially made for the study: a cardboard mockup and a computer simulation programmed in Microsoft's VisualBasic.

2.2.1 The cardboard mockup.

The cardboard mockup (figure 4) was made by the first author, in cooperation with the Industrial Design Engineer who conducted the user tests (the facilitator). It took about one person day and a half to build the mockup. The product's case was simulated by cardboard boxes. Inserting a tape was simulated by laying it on top of the box. On the front side of the box, small ridges were made for placing cards representing menus, TV images, LEDs and the counter/clock. For values within menu-items, Post-It® Notes were used. Changes of menu, changes within menus and changes of feedback or TV images were done manually by the facilitator. Menu cards, TV images and feedback cards were prepared in advance, the Post-It® Notes were written when needed. The RC was simulated by a wooden block (about real size) with color prints of the 'daily' and 'special' sides of the RC glued on it.



Figure 4. The cardboard mockup (showing the Timer for today menu) with cards and Post-It® Notes as feedback.

2.2.2 The software prototype

The software prototype (figure 5) was made by Paremion, a company specializing in multimedia software productions, who needed 60 person hours to complete the prototype. On the 19-inch computer monitor, one side of the remote control (RC) was shown next to a simulated TV screen. Near the lower edge of the screen two extra (toggle) buttons were made: one for changing the view on the RC (for switching 'daily side'/'special side') and one to simulate inserting or ejecting a tape. TV-images and recorded images were simulated by animations of a walking 'stick figure' against various backgrounds. Different channels showed different animations. When switching channels, it took about two to three seconds for animations to appear; the indication for channel numbers (or the indication 'tape' in the top-right corner of the TV screen) would appear almost instantaneously. The tree of menus and settings was structured like the menu of the real product. Menus and feedback in the form of LEDs and on-screen messages were similar to those of the real product with respect to their contents. Functionality behind menu settings was simulated for functions directly related to watching the TV, as well as for using (playing, winding etc.) and programming the video.

What's the problem?

Studies on identifying usability problems in user tests



Figure 5 The software prototype (showing the picture menu). Both TV/video recorder and remote control are shown on the computer screen. The large buttons along the lower edge of the screen are named 'Turn over remote control' and 'Insert cassette'.

Due to insufficient screen size and/or screen resolution, the labels on the RC were hard to read. Therefore, participants received an A4-sheet showing the RC with labels in a more readable form. For controlling the TV-video combination, participants were asked to point at buttons shown on the screen. Subsequently, the facilitator would use the mouse to click the button the participant pointed to. Due to difficulties in programming the simulation, there were some small inconsistencies in the prototype, (e.g., the time counter did not stop at '00:00' when rewinding a tape; functions like 'play' and 'rewind' were not blocked when a program was being recorded).

2.3 Test set-up, test procedure, participants

2.3.1 Participants

The test was performed by 18 Dutch participants (9 male, 9 female) who were equally divided over the three conditions (i.e., product, cardboard mockup, software prototype). In each condition, there were two participants from each of the following three age categories: 15-18 years, 30-40 years, 60-70 years.

2.3.2 Test set-up

In the period 1996/1997 all tests were conducted in the user interface laboratory of the Faculty of Industrial Design Engineering at Delft University of Technology. In case of the product and cardboard mockup, participants sat behind a table at a distance of

about 4 m from the TV screen. For using the computer simulation, participants sat at a table directly in front of the computer. In front of the participant, on the table were: the RC, a (supposed) TV guide with information about the channels 1, 2 and 3 for 'today' and 'tomorrow', the Dutch language part of the user manual, and a reference card briefly explaining the functions of buttons on the RC. All sessions were video recorded using three cameras: two focusing on the table, hands and remote control and one focusing on the TV-screen; in case of the software prototype, two cameras focused on the screen. Figure 6 depicts a schematic overview of the test setup for the product and the cardboard mockup.

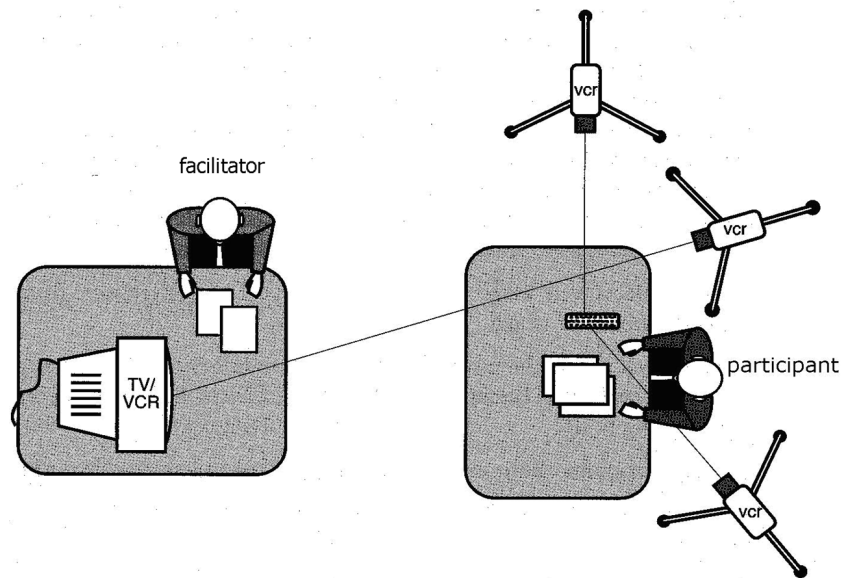


Figure 6 Schematic overview of the test-setup for the tests with the product and the cardboard mockup.

2.3.3 Test procedure

The procedure for conducting a session was as follows: participants were told that the test concerned the use of a combined TV video recorder and that the facilitator would ask them to perform some tasks described in task scenarios, after they had explored the use of the product for a maximum of 5 minutes. Participants were asked to talk aloud as much as possible, and to especially verbalize what actions they intended to perform. In addition, they were asked not to withhold utterances of surprise or frustration in case something unexpected would happen. They were asked not to explain things to the facilitator, but to talk aloud to themselves, as if they were alone in the room. Before participants started to briefly explore the use of the product, they were

What's the problem?

Studies on identifying usability problems in user tests

instructed about some specifics related to using the product or prototype in the test situation. Examples of such instructions are: for the real product: 'there are no real broadcasts', 'teletext does not function'; for the cardboard mockup: 'we act as if this box is the TV-video recorder and this thing is the RC. Just press buttons and speak out loud which button it is, so that I know which button you pressed'; for the computer prototype: 'this is a special RC, it has two sides. You can change side by pressing this button' and 'you don't have to use the computer yourself, just point to the button you want to press, I will take care of the necessary actions'.

2.3.4 Tasks and task scenarios

Fourteen participant tasks were read out loud by the facilitator and were repeated on request (or spontaneously if it was clear that the participant had forgotten the task). The facilitator was not allowed to help the participant (only encouraging or reassuring was allowed) and was instructed to stop task performance if a participant after some time of trying was unable to perform it or seemed to get too distressed. In many sessions some tasks were cancelled beforehand, because there was no need to perform them any more. For example, a number of the tasks were devised just because the product needed to be brought to a certain status in preparation for a subsequent task. However, sometimes participants had already brought the product in the required status and therefore there was no more need to do that anymore. In most cases, this concerned simple tasks like switching channels. Tasks 2, 3, 4, 5, 8, 10, 11 and 14 (see table 1) were performed by all 18 participants. These were used as the basis for making comparisons. The time needed to perform all tasks ranged from 15 to 45 minutes. The crossed cells in figure 7 indicate the tasks that were not performed by the specified participants. Table 2 shows example task scenarios of task 5 and 14.

Table 1 The tasks that were performed by all participants

-
2. Lower the sound volume
 3. Make the screen image less colorful
 4. Have a look what's on the tape
 5. Position the tape exactly at the beginning of the recorded movie.
 8. Rewind to the very beginning of the tape
 10. See what's recorded at the beginning of the tape.
 11. Pause the tape, so that the image freezes.
 14. Schedule the recorder so that it records a specific TV program tonight.
-

Mockup			Task	1	2	3	4	5	6	7	8	9	10	11	12	13	14
P1	male	15-18			3	1								5	X	X	1
P2	female	15-18		1	3	1			3					3	X	X	12
P3	male	30-40	1	1	5				X	X				7	X	X	6
P4	female	30-40	3	1	6									4	X	X	8
P5	male	60+	1	5	7		2		1	2	3			1	X	X	6
P6	female	60+		5	6	1				1	1			11	X	X	5

Product			Task	1	2	3	4	5	6	7	8	9	10	11	12	13	14
P7	female	15-18			8									14	X	X	4
P8	male	15-18			2	2			1	1				1	X	X	5
P9	male	30-40	2		1									6	X	X	10
P10	female	30-40			2									7	X	X	9
P11	female	60+			8	7	11				12	1	3	15	X	X	30
P12	male	60+	X	1	1	1			1	1	4			5	X	X	3

Software			Task	1	2	3	4	5	6	7	8	9	10	11	12	13	14
P13	female	15-18	2		5					1			1	6	X	X	6
P14	male	15-18		1	3		8							4	X	X	4
P15	male	30-40	3	1	3									1	X	X	4
P16	female	30-40		10	2	11							1	2	X	0	8
P17	female	60+	1	4	9	6	2		1	7			1	3	X	X	14
P18	male	60+	X	3	11	4			2	1		X		5	X	X	16

Figure 7 Overview of task performance per participant for each of the three conditions. Grey cells indicate unsuccessful task outcome (task goal not achieved), crossed cells indicate that the task was not performed by the participant. The figures within the cells indicate the number of difficult moments, based on the DEVAN analysis (for details see Results).

Table 2 Example task scenarios of tasks 5 and 14

Task 5 What you currently see on this tape are some short commercials. The commercials are followed by a movie that you have recorded before. Please try to position the tape exactly at the start of the movie.

Task 14 Tonight you won't be at home. However, you would like to record the program 'Chirurgenwerk' (Channel 2, 22.12-22.16hrs). Do you think you can record the program while you are not at home? Could you please try and do that?

2.4 Processing the recorded sessions using DEVAN

The video recorded sessions were processed using the video analysis tool DEVAN (Vermeeren et al. 2002, see chapter 2). This tool was developed to improve an analyst's consistency in the extraction of usability problems and to make that process explicit. For that, it prescribes a detailed data analysis procedure as well as a checklist that defines interaction event types that signal difficulties in interactions. Interactions are

What's the problem?

Studies on identifying usability problems in user tests

transcribed to a specified format assisting analysts in making explicit how they interpreted interactions (figure 8 and 9).

Two main stages are distinguished in the analysis, consisting of three and two sub stages, respectively:

- 1 *Creating a table that represents the interaction at multiple levels of abstraction:*
 - *Transcribing actions, verbal utterances and non-verbal behavior.*
 - *Preliminary segmentation of the interaction based on threshold pause times between actions.*
 - *Defining interaction segments, as well as clustering and abstracting these to (sub) task-level episodes.*
- 2 *Creating a list of difficult moments in the interaction:*
 - *Detecting events that signal a difficulty, by using a checklist of signal event types.*
 - *Describing the observed difficult moments.*

time stamp	log	interaction segments	context	breakdown indications
0:00:00	action	description of interaction segment	task 1 description	code
0:00:12	action			
0:00:13	action			
0:00:19	action	description of interaction segment	task 2 description	code
0:00:21	action			
0:00:22	action			
0:00:23	action			
0:00:24	action			
0:00:25	action	(verbal utterance)		
0:00:44	action	description of interaction segment	task 2 description	code
0:00:46	action			
0:00:49	action			
0:00:50	action	(verbal utterance)		

1 column for logging user-product interaction 2 primary boundary, indicating the start of a new interaction segment 3 secondary boundary, indicating the possible start of a new interaction segment 4 column for interaction segment boundaries and descriptions 5 column for task descriptions and descriptions of intermediate level episodes 6 column for breakdown indication type codes 7 event marked as breakdown indication

Figure 8 General format for DEVAN's interaction overview table. NB. 'Breakdown indication' is a synonym for 'signal event'.

Time	Actions	Interaction segments <i>Utterances</i>	Tasks	Signal codes
12,04	Task 14		Task 14 Record "a Surgeon's job" tonight from 21.22 - 22.16, Netherlands 2	
		<i>let's try... I am not sure whether I will succeed in this... I can try</i>		
12,10	menu today	evokes <menu today> <i>let's have a look, first time for today... starting time... have a look...I want to go down...</i>		
12,24	cursor,down	moves to item <starting time> <i>starting time... what's the name of the programme again? facilitator: a surgeon's job participant: a surgeon's job... Netherlands 2... 21.22... starting time 21</i>		act
12,42	number,2	sets starting time and		
12,42	number,1	then moves to stop time		
12,43	number,2	22		
12,45	number,2			
12,47	cursor,down	<i>stop time</i>		
12,50	cursor,up	changes plan and first goes to menu item <programme> to set it to 2		corr
12,51	cursor,up	<i>Oh, wait... it's on 2 wasn't it?</i>	rec	
12,53	number,2	<i>(verifies in TV guide)</i>		
13,00	cursor,down	moves to item <stop time>		
13,00	cursor,down	<i>stop time... surgeon's job... 2216...then I set it to 2230</i>		

Figure 9 Example of an interaction overview table (participant 12, product condition – translated from Dutch original).

At the end of stage one, the interaction is represented in the format shown in figure 8 (except for the grey marks (figure 8, item 7) and the signal event codes (figure 8, item 6), which are added in stage two). The interaction table includes all recorded actions, as well as transcriptions of utterances and non-verbal behavior that are used as the basis for detecting difficult moments. The segmentation in combination with the abstractions makes explicit how analysts interpret a participant's interaction with the product. The checklist of signal event types (table 3) serves as a list of event types that assist in recognizing difficult moments. Identified signals of difficulties are then listed and described using the following elements: 1) a time code reference, 2) a description of the

What's the problem?

Studies on identifying usability problems in user tests

observed behavior, 3) the context in which the event occurred (task context and product mode), 4) the code for the type of signal event, 5) a free-form description of the signal event. It should be noted that a difficult moment can be signaled by multiple signal event types at a time. For example, an analyst may observe that the user selects the wrong button (first signal), may hear the user say “oops, that was wrong” (second signal) and may then see that the user undoes the wrong action (third signal). Therefore, multiple events signaling the same difficult moment have to be grouped before comparisons between conditions can be made. The outcome of a DEVAN analysis is an uncategorized list of difficult moments in interactions.

Table 3 DEVAN’s checklist of signal event types (abbreviated descriptions; adapted from Vermeeren et al., 2002).

Signal event types based on observed actions on the product		Signal event types based on verbal utterances or on non-verbal behavior	
ACT	User chooses wrong action	GOAL	User formulates an inadequate goal
DISC	User discontinues an initiated action	PUZZ	User’s seems to be puzzled about what to do next.
EXE	User has problem in physically executing an action	RAND	From the user’s words it is clear that actions are selected at random.
REP	An action is repeated with exactly the same effect.	SEARCH	User indicates to be searching for a specific function and can’t find it, or function does not exist.
CORR	User corrects or undoes a preceding action.	DIFF	User indicates that physical execution of an action is problematic or uncomfortable.
STOP	User stops task, task not successfully finished.	DSF	User expresses, doubt, surprise or frustration after having performed an action
		REC	From the user’s words it is clear that a preceding error is recognized as such, or that something previously not understood now has become clear.
		FBK	From the user’s words it is clear that some feedback has not been perceived or correctly understood
		QUIT	User realizes that the current task was not successfully finished, but continues with next task.

What's the problem?

Studies on identifying usability problems in user tests

participant's language problems explained a lot of the difficult moments the person experienced (e.g., not realizing having to press the play button in order to play a video tape). As the aim of the case study was to compare findings from the three situations and trace back differences to characteristics of each, homogeneity in groups was needed. Therefore, in part of the subsequent analyses the data from this participant are dealt with separately.

3.2 Detailed analysis 1: Signal event types

3.2.1 Action vs. expression based signal events

The findings so far are largely in line with those reported in the literature: summative high level measures of task performance (i.e. task effectiveness and counts of difficult moments) hardly differentiate between tests with products and tests with various kinds of prototypes. However, this does not necessarily mean that no differences in behavior or performance exist. Various effects on a more fine-grained level may compensate each other, thereby leveling out any differences in summative measures. Therefore, a more detailed analysis based on identified signal events will now be discussed.

The various types of DEVAN signal events can be divided into two broad categories: those based on the user's actions on the product and those based on the user's (verbal or nonverbal) expressions. Figures 11a and 11b provide overviews of the number of 'action' based and 'expression' based signal events per participant. Note that for each participant the number of signal events not necessarily equals the number of difficult moments, because a single difficult moment can be recognized based on multiple signaling events (e.g., observing a wrong choice of action and hearing an utterance indicating frustration). The participant showing an extreme number of signals (in the product situation) is the same person as the outlier shown in the overview of difficult moments (participant P11).

The general pattern seems to be that task performance on the real product led to slightly more action-based difficulty signals than task performance on the prototypes. For the expression based signals the opposite is found; especially for the software prototype, the number of expression signals seems to be higher. But again, differences are negligible.

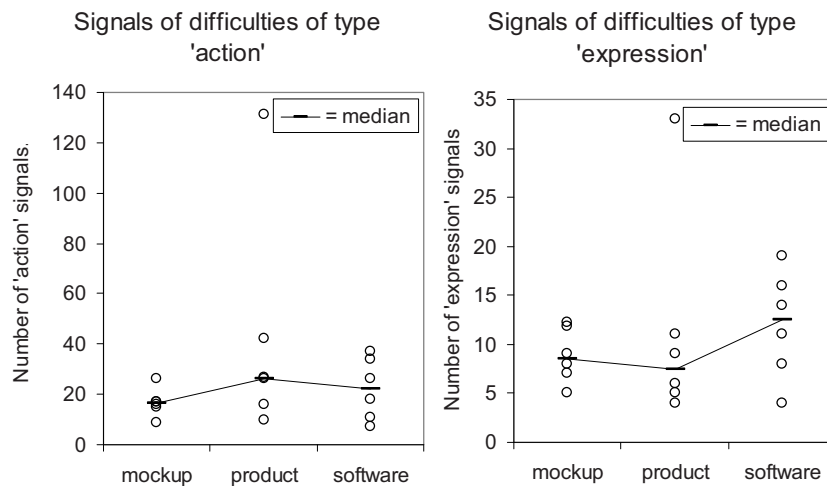


Figure 11a Number of signal events of type 'action' in each of the three conditions (each dot represents one participant)

Figure 11b Number of signal events of type 'expression' in each of the three conditions (each dot represents one participant).

3.2.2 Signal event types

To further analyze which difficulty signals were found in each of the three conditions, figures 12a and 12b provide overviews of the total number of signal events per signal event code. The data have been averaged across participants and are shown per condition. Figures 12a and 12b provide an overview of the action- and expression-based signal events, respectively. Note that the person with the extreme number of signals and difficult moments (P11) has been left out of these overviews and will be dealt with separately.

Action-based signal types Figure 12a shows that in all three conditions, the ACT signal event type ('wrong choice of action') was found most. Furthermore, using the product led to more signal events of type CORR (corrections of errors) and REP (repetition: immediate and exact repetitions of wrong actions, with exactly the same effect) than using the prototypes. Including P11 would result in almost doubling the average number for ACT and REP signals in the product situation (grey bars in figure 12a).

What's the problem?

Studies on identifying usability problems in user tests

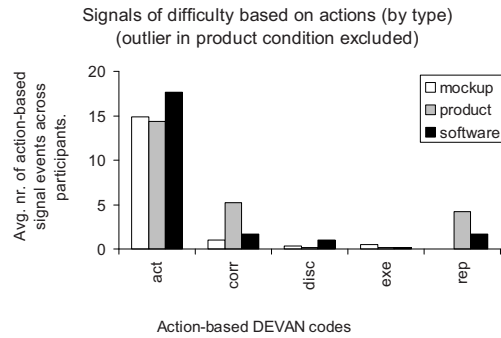


Figure 12a Distribution of action signal types for each of the conditions. Shown are the average numbers across participants.

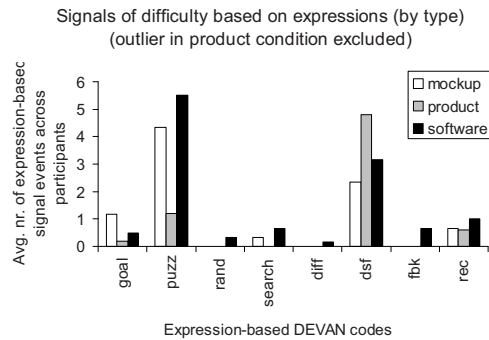


Figure 12b Distribution of expression signal types for each of the conditions. Shown are the average numbers across participants.

Further inspection of the data shows that differences in corrective actions (CORR) and in immediate repetitions of errors (REP) were mostly found in tasks 11 and 14. Table 4 provides an overview of the number of CORR and REP signal events in tasks 11 and 14.

Table 4 Overview of numbers of CORR and REP signal events in tasks 11 and 14.

	CORR			REP		
	Mockup	Product	Software	Mockup	Product	Software
Task 11 Video still image	1	11	4	0	15	0
Task 14 Timer recording	2	7	3	0	2	8

In task 11 participants typically first (erroneously) tried the pause/stop button for getting a still video image, noticed the ineffectiveness of their action, and some participants then pressed the record button. This is not as illogical as it seems, because the 'pause/stop' label was in the middle between the actual pause/stop button above it, and the unlabeled red record button below it. As a result of pressing the record button, in the product condition, the product started recording and most other functions were then blocked (usually without the participant noticing it). Participants trying to use the blocked functions did not get any product feedback when doing that. They then wondered why the product didn't react and repeatedly pressed (a REP signal) the blocked functions (as previously they had successfully used them and they didn't understand why they didn't function anymore). Subsequently they (often unsuccessfully) tried to correct the situation (hence the CORR signals). This led to participants concluding that the product didn't function anymore, to participants giving up on the task and to sometimes very lengthy task episodes.

In task 11 of the mockup condition the record function was erroneously used as well, but the frustrating situation observed in the product condition was not found. There were several reasons for the non-occurrence of blocked functions (and thus of the absence of REP signals), some of which related to the fact that a facilitator mediated the interaction between participant and mockup:

- 1 In two cases there was a communication problem between facilitator and participant: the participant wanted to press the record button (the wrong action), said “press STOP”, and pointed at the record button thinking that was the stop button. In response to that the facilitator, not being able to exactly see what button the participant pointed at, acted as if the real stop button was pressed and thus inadvertently avoided the blocking of functions and subsequent repeated button presses. This could easily happen because the label ‘pause/stop’ (which was mentioned by the participant) was just below the actual pause/stop button and just above the unlabeled record button (that the participant intended to use).
- 2 In two other cases the facilitator also did not block the use of other functions after the participant had pressed the record button. Most likely because the facilitator herself at that moment did not realize she had to block all other functions.

The miscommunication between facilitator and participant mentioned above occurred in the software condition as well. In addition, several other situations were observed in which characteristics of the software prototype or the presence of the facilitator prevented the blocking of functions (and thus the REP signal events) to occur:

- 1 In one case the participant wanted to press the record button, but then recalled the command before the facilitator had activated the record button. Mediation by the facilitator made this possible; most likely with a functioning product the button would have been pressed immediately.
- 2 In another case the participant intended to simultaneously press the record button and another button (which is technically impossible in the software prototype, because there is only one mouse cursor).
- 3 Blocking functions was not fully implemented in the software prototype because it was too time consuming to implement.

In sum, in case of task 11, both in the mockup condition as well as in the software condition participants were often not confronted with blocked functions. Thus, the situations in which participants ended up in lengthy interaction episodes repeatedly pressing non-functioning buttons thinking the product didn’t function anymore were not observed in the prototyping conditions.

In task 14 (scheduling a timer recording) qualitative analyses did not reveal any situations in which characteristics of prototypes led to differences in interactions clearly relating to CORR signals. For the REP signals this was different. In the software condition task 14 resulted in eight repetitive (REP) actions, all but one from one single participant. This participant was trying to set a value for a menu-item. While doing that the participant did not notice that the cursor bar was not on the item for which (s)he was trying to set the value or did not understand that it needed to be there. After the initial REP actions, the facilitator tried to point this problem out to the participant several

What's the problem?

Studies on identifying usability problems in user tests

times, but did not succeed (and the participant kept on trying). Presumably, the problematic situations that led to the REP actions in task 14 of the product and software condition are less likely occur in the mockup condition, because in that situation feedback is much more obvious, as it requires the facilitator to perform very noticeable physical actions.

Expression-based signal types Figure 12b shows that the product sessions yielded more DSF signal events (DSF: doubt, surprise or frustration after seeing the effect of an action) than the prototype sessions, whereas for the PUZZ signal (puzzled: user does not know exactly what action to choose) the opposite was found. Including P11 would lead to roughly a doubling of the number of PUZZ signals and to eleven RAND signals (random: actions indicated by the user as being random) in the product condition. In the graph in figure 12b the eleven RAND signals would appear as an average value of 1.8. PUZZ events are events in the form of (non) verbal user expressions that take place during the *preparation* of actions, i.e. during the process of forming intentions on how to proceed. In an analysis of the contents of these expressions two types of expressions could be distinguished: those that contain clues as to what product properties or characteristics are considered in the intention forming process and those that don't. Examples of utterances containing such clues are: "how do you switch it to video mode?" "maybe that little cross?" "I need to find something that can go to the left". Examples of utterances containing no such clues are: "I can't find it..." "I have no idea, let's have a look in the manual" "let's press sound and then see what happens" (NB. this counts as 'no clue' as it doesn't provide any product-related information in addition to the actions on the product that can be readily observed). Table 5 denotes how the two types of expressions are distributed across the three conditions.

Table 5 PUZZ events providing additional insight into relevant product characteristics.

PUZZ events	Mockup	Product	Software
Event provides additional clues as to what product characteristics or properties are considered	19	2	14
Event provides <u>no</u> additional clues as to what product characteristics or properties are considered	7	4	19

In the mockup sessions relatively more utterances containing useful product-related clues were found than in the other sessions (mockup: 19, product: 2, software: 14). The fact that in the mockup condition participants had to communicate their intended actions to the facilitator and had to wait longer for product responses (simulated by hand by the facilitator) may have played a role in the PUZZ events being more informative. DSF events take place during the process of interpreting and evaluating a product's response *after* the user has performed an action. In an analysis of the contents of these

expressions, three types of expressions could be distinguished: 1) those that make clear what product properties the participant is unsure, surprised or frustrated about, 2) those that make clear how participants interpret the product's or prototypes' responses to actions or what their assumed reasons for the responses are, and 3) those that provide no specific indications other than that there is doubt, surprise or frustration. Examples of utterances that indicated what product properties made participants unsure, surprised or frustrated are: *"all the time I have been turning the rotary and still..."*, *"that's strange... I wanted to stop the video and still see the image..."*, *"so... but... if you press pause... you should get a still image..."*. Examples of events that make clear how participants interpreted responses, or what they assumed to be reasons for responses are: *"so... the upper one means... to the left more sound than to the right..."*, *"so the number 1 indicates the channel... or doesn't it..."*, *"oops, it is not functioning anymore"*. Examples of unspecified utterances of doubt, surprise or frustration are: *"that's strange!"*, *"huh?"*, *"it's gone!"*. Table 6 denotes how the three types of expressions are distributed across the three conditions. No striking differences were found between conditions with respect to the type of information the utterances contain. In all conditions, more than half of the DSF signals contained useful information.

Table 6 Three categories of DSF events.

DSF events	Mockup	Product	Software
Events that make clear what properties or characteristics of the product's response the user is unsure, surprised or frustrated about	6	6	8
Events that make clear how a participant interprets the response, or their own reasoning of why they think they received this response	3	8	6
Events that provide no specific indications other than that there is doubt, surprise or frustration	5	10	5

3.3 Detailed analysis 2: Error taxonomy

As a second way of analyzing differences between conditions in more detail, difficult moments as identified by the DEVAN tool were categorized using the taxonomy of usability errors developed by Zapf et al. (1992). This taxonomy was used because of its ability to relate usability problems to characteristics of user behavior. It was developed from an action-theory perspective where...

"... mismatches of usability can be differentiated according to steps in the action process and different levels of action regulation.

What's the problem?

Studies on identifying usability problems in user tests

Within this approach, the action process comprises goal and plan development, the execution of actions, as well as monitoring, and feedback processes.

... three levels of action regulation are distinguished within the framework of hierarchically (or better, heterarchically) organized action plans and goals". (Zapf et al., 1992)

On the knowledge-based level, regulation is conscious, predominantly works in a serial mode, interpreting feedback step-by-step. On this level conscious reasoning processes take place. On the rule-based level, actions are regulated by ready-made action patterns available in memory that are triggered by situationally defined parameters; such processes can be regulated consciously, but they need not necessarily be conscious. At the sensorimotor level stereotypes and automatic movement sequences are organized without conscious attention. Regulation takes place with the help of proprioceptive and exteroceptive feedback and is largely unconscious. In addition to the levels of action regulation the framework defines a knowledge base for regulation that can contain knowledge of facts and procedures and understanding in the sense of mental models. In the framework this knowledge is assumed to be used for developing goals and plans. Zapf et al. (1992) emphasize that understanding the two dimensions (i.e., action regulation levels, and steps in the action process) is very important for understanding the errors. Figure 13 shows how the taxonomy is organized.

Knowledge base for regulation			
Knowledge errors			
Action regulation levels	Steps in interaction process		
	Goal/planning	Monitoring	Feedback
Knowledge-based level	Thought errors	Memory errors	Judgment errors
Rule-based level	Habit errors	Omissions	Recognition errors
Skill-based level	Sensorimotor errors		

Figure 13. Error taxonomy, adapted from Zapf et al. (1992).

Figure 14 shows the number of difficult moments categorized according to Zapf et al.'s (1992) error taxonomy. Again participant P11 is left out of the overview. Inclusion of P11 would lead to roughly a doubling of the knowledge problems and judgment problems in the product situation.

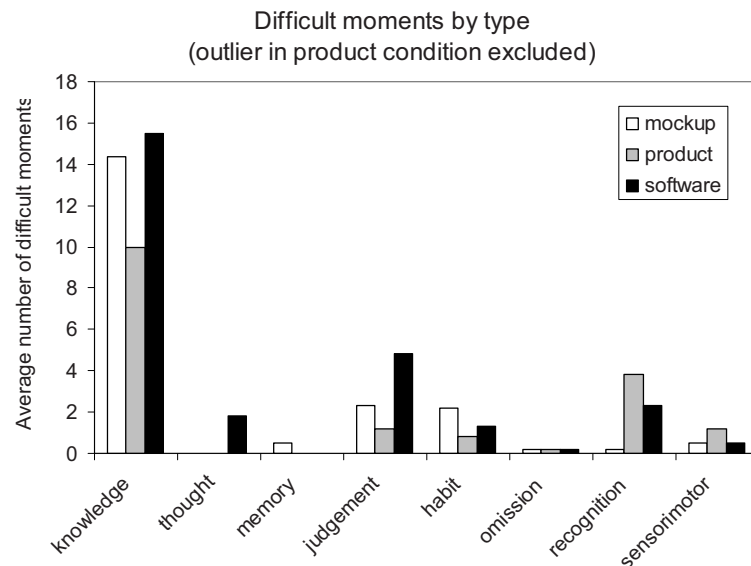


Figure 14 Difficult moments by type as defined in the error taxonomy of Zapf et al. (1992).

3.3.1 Knowledge errors

Figure 14 shows that the majority of difficult moments relate to knowledge errors. Knowledge errors are errors that occur if participants have difficulty performing their tasks, because they do not know certain actions, buttons, procedures and so forth. This is different from for example thought errors, where participants develop inadequate plans or take wrong decisions in assigning plans and subplans even though they know all the required features of the product. As the current study is about first-time use of a product, it is not surprising that the majority of the difficult moments are caused by a lack of knowledge about the product's functioning and use. The average number of knowledge errors was higher for the prototypes than for the product. A closer look reveals that the differences in knowledge errors predominantly occurred in task 2 (mockup: 12, product: 1, software: 18). In task 2, participants were asked to change the sound volume. It is striking that this task hardly caused any difficulties in the product sessions, and a substantial number of problems in the prototype sessions. A combination of two factors seems to have caused the difference in conditions: 1) the two-sidedness of the remote control, and 2) the fact that participants were asked to explore the TV-VCR for a maximum of about 5 minutes before starting the tasks.

Two sidedness of the RC In the mockup sessions, two participants assumed that the so-called 'daily' side of the RC with the play and (re)wind buttons was especially for video functions, and not for TV related functions; as the volume buttons were on that side as

What's the problem?

Studies on identifying usability problems in user tests

well, they concluded that they shouldn't use those for changing the sound volume. While looking at the 'special' side of the RC another participant had forgotten that it had two sides (and therefore didn't find the volume buttons). The fact that the buttons on the mockup RC could not be felt (they were buttons printed on paper) may also have played a role here. In the software session, two participants indicated that they had forgotten about the other side of the RC as well. Three others tried to use the cursor keys at the 'special' side of the RC instead of the volume buttons on the 'daily' side. It is likely that they also had forgotten about the back side of the RC. The fact that the RC was not hand-held (it was depicted on the screen) and had to be turned over by pressing a dedicated button is likely to have played a role here.

Five minutes of exploration The other reason for the differences in problems in task 2 may lie in the five minutes of exploration prior to task performance. In all three conditions participants were invited to explore the TV-video combination for a maximum of five minutes. In the product sessions, participants usually did not use the full five minutes, but they at least did some exploration; in the prototype sessions they hardly did. During the explorative phase in the product sessions, one of the first things participants generally did was trying out the rotary which was prominently visible on the RC. They then discovered that by using the rotary they could switch between TV channels. By doing that they implicitly learned that that side of the RC could be used for TV functions, whereas some participants in the mockup sessions (who had not explored the product and therefore had not already tried the rotary) thought it was only for video functions. Most likely, the main reason for the lack of exploration in the prototype sessions is the fact that participants could not directly interact with the prototype, but had to instruct the facilitator what to do. This makes it less likely that participants start trying various buttons (like the rotary) just to see what they do. A possible implication of this is that explorative behavior is difficult to evoke in prototyping situations. In this case, this lack of exploration led to observing problems that did not occur in the use of the real product.

3.3.2 Judgment errors

Figure 14 also shows that a considerable number of difficult moments could be categorized as judgment errors and that there were differences between product sessions and prototype sessions. Judgment errors occur if a participant has problems understanding or interpreting the product's feedback after an action.

Remarkably, in the software condition there was one specific problem that led to a relatively large number of judgment errors in task 14. The problem was that the VCR menu was shown on the simulated TV screen, which in turn was depicted next to the RC on the computer screen. Because of that participants sometimes got confused on how to control the menu. Being used to a computer they wanted to press a menu-item directly on the depicted TV screen, instead of pressing a button on the depicted RC. This was a recurrent problem in the software condition.

3.3.3 Recognition errors

Finally, recognition errors also showed an interesting pattern for further analysis. Recognition errors occur when the appearance of feedback information is not noticed or where (on a rule-based level) this information is confused with other feedback information. Figure 14 shows that recognition errors hardly ever occurred in the mockup sessions (only one time in the mockup condition), were more often found in the software sessions and even more in the product sessions. Especially the difference between the mockup and the product condition is striking.

In most cases recognition errors were related to problems of not noticing the feedback that indicated the current status of what the VCR was doing with the tape (e.g., not noticing it is winding, not noticing the tape has stopped, not noticing it is recording). This happened mostly in product sessions, but also in software sessions (mockup: 0, product: 13, software: 8). Feedback on what the VCR is doing with the tape is given in the form of LEDs above the TV screen (or simulated LEDs above the simulated TV screen in case of the software prototype, or cardboard cards placed above the cardboard TV screen in case of the mockup). In addition, in case of the product the mechanics for winding the tape may be heard (and no sound was heard in case of the prototype conditions). In the mockup condition, a change in color of a simulated LED does not go unnoticed, as it is simulated by the facilitator manually placing another card above the simulated TV screen. This is such a noticeable action that participants never missed it. This explains the difference between the mockup sessions on the one side and the product and software sessions on the other. For this product in particular the fact that these problems were missed is important, because of this product's specific mechanisms for the rewind and fast forward functions which deviate from what is usual in VCRs (see section 2.1).

Another reason for the difference in recognition errors between the mockup sessions on the one side and the product and software sessions on the other is that participants sometimes did not notice that a just entered value for a menu-item was automatically reset by the product if it was not confirmed by pressing OK before proceeding to the next menu-item (mockup: 0, product: 4, software: 3). In the mockup condition automatically resetting a value after not confirming it was simulated by manually removing a Post-It[®] Note and putting back the one that was just taken off. This action was much more noticeable than a changed number at a place on the screen that the attention had just moved away from. Therefore, recognition errors of not noticing the automatic resetting of a value did not occur in the mockup condition.

4 Conclusion

Findings from usability tests on a combined TV-video cassette recorder were compared to those from usability tests with a cardboard mockup and a computer software prototype of the same product. The DEVAN tool for extracting difficulties from observations was used to improve consistency in the analysis and to make it possible to inspect the

What's the problem?

Studies on identifying usability problems in user tests

data at various levels of abstraction. This made it possible to use a theory-based error taxonomy for classifying the difficult moments in the observed interactions. High-level task performance measures (task effectiveness and counts of difficult moments) showed little differences in findings between conditions. However, a more detailed analysis showed a variety of differences in how participants behaved and in types of identified problems. It should be noted that in this case these differences related to important interaction designs that were very typical for this specific product. Thus, in a product development situation these would be the kind of problems designers would be interested in. Many of the differences are related to the fact that in the sessions with prototypes a facilitator mediated the interaction between participant and prototype. This proved to have both advantages (e.g., more informative verbal utterances), as well as disadvantages (e.g., less exploration, fewer series of repeated, ineffective button presses). In addition, some typical characteristics of prototypes had an effect on the types of usability problems that were found. Characteristics that were found to be relevant included: incomplete implementation of functionality, representing a three-dimensional RC in two-dimensions on a screen, showing an RC for controlling menus and the related menus on the same screen, changes in feedback being more noticeable in a cardboard mockup).

Although the product used in this study may be outdated, we believe that the findings from this study are still relevant. If we would have stuck to comparing high-level measures like task effectiveness and counting difficult moments our findings would not have been generalizable beyond this exact product and the exact prototypes that were used. However, in the detailed analyses we were able to track down how more general characteristics of prototypes and prototype usage can influence participant behavior and the types of usability problems identified. This makes our findings potentially useful to situations beyond those with exactly the same product or prototypes.

5 Practitioner's take-away

- When comparing usability evaluation methods based on high-level task performance measures like task effectiveness and problem counts only, relevant differences between outcomes of methods will be missed.
- Using a facilitator that mediates the interaction between participant and a prototype can have various effects on findings:
 - In case of cardboard mockups, participants' utterances may contain more clues as to product characteristics that play a role in their mental processes when preparing for an action.
 - Participants may be less inclined to freely explore product use and try out unconventional functions. This lack of exploration may lead to identifying problems in task performance that otherwise are less likely to occur.

- In case of a cardboard mockup, participants may not show the typical behavior of repeatedly pressing a button that does not appear to cause any observable response (e.g., due to unnoticed, subtle feedback or a lack of feedback).
- In case of a strictly verbal mediation, problems that are due to confusion about which label belongs to which button may be less easily found due to miscommunication between participant and facilitator. If participants are asked to clearly point at the buttons they intend to use and this pointing can easily be observed by the facilitator, this problem is less likely to occur.
- Some actions may bring a product in a mode that blocks the use of many other functions (e.g., the record function on a VCR). Incomplete implementation of this blocking effect makes it difficult to find out to what extent participants can recover from such actions. In this study the impact of such an erroneous action on further use was often very high.
- If a software prototype depicts a manual menu control device (e.g., remote control) and its related menu on a single computer screen, this can lead to confusion in controlling the menu; participants may try to control the menu by direct pointing instead of by pointing at the buttons on the depicted control device. This can be a recurring problem.
- The use of both sides of a two-sided manual control device (like a two-sided remote control) may become more difficult if the user can not feel that it is two-sided (e.g., if buttons can not be felt) or (in case of a software prototype) if only one side is displayed on a screen and it requires a button press to show the other side.
- In a cardboard mockup, subtle product feedback mechanisms like a LED changing color or automatically resetting the value of a menu-item if it is not confirmed, are often simulated in a way that makes them more noticeable than with a real product or a software prototype. Thus problems that are due to unnoticed subtle feedback may be missed with a mockup.

What's the problem?

Studies on identifying usability problems in user tests

chapter 5 Discussion and conclusion

Main objectives of the study

Analyze consistency in user test data analyses and devise a way of dealing with inconsistencies, by:

- 1 Developing a data analysis procedure that makes the analysis documentable, that allows tracing back of usability problems to the original observations and that provides insight into the process of transforming data from observations into usability problems.
- 2 Studying to what extent the developed procedure exposes possible causes of inconsistencies and reduces less persistent differences (i.e., those caused by fatigue, lack of vigilance and distraction).
- 3 Applying the procedure to find out whether it makes the data from the comparative 'prototypes' study inspectable, so that differences in identified usability problems can be traced back to characteristics of the prototyping situations.

What's the problem?

Studies on identifying usability problems in user tests

1 Developing the (Slim) DEVAN data analysis procedure

Chapter 2 reported the development and evaluation of DEVAN, a procedure for improving consistency in user test data analyses and for making these inspectable. To allow for inspection, DEVAN makes use of tables representing interactions at various levels of abstraction. Results from the various steps in the analysis refer back to data elements in these tables. Consistency in creating the overview tables is facilitated by segmenting interactions using threshold times for pauses between actions. A checklist of observable signal events helps analysts remaining consistent in detecting when a usability problem occurs. Difficulties in interactions are described using a standard format containing references to specific elements in the interaction overview tables.

Chapter 3 reports the development of an adaptation of DEVAN named SlimDEVAN. Because the use of DEVAN led to session time/analysis time ratio's that were unacceptably high for the studies reported in chapter 3, SlimDEVAN was developed as a simpler, less time-consuming version of DEVAN. Its interaction overview tables lack the multiple levels of abstractions and it has no prescribed way of segmenting interactions. The checklist of signal events is used in the same way as in DEVAN, but difficulty reports are substituted by usability problem reports. The report formats are simpler than in DEVAN, while still containing the necessary references to elements of the interaction overviews.

2 (Slim)DEVAN, inspectability and consistency

In chapters 2 and 3, various issues of inspectability and consistency were studied by using (Slim) DEVAN. In chapter 2 DEVAN's inspectability was tested by comparing multi-

What's the problem?

Studies on identifying usability problems in user tests

ple lists of difficulties extracted from the same videotaped test sessions. The lists were created by two analysts (the developers of DEVAN) working independently. In chapter 3 inspectability and consistency were studied by comparing problem reports from three labs that had tested the same product using a test protocol they developed collaboratively. Each lab had analyzed their data twice: first using their own regular data analysis procedure and subsequently applying SlimDEVAN. In a follow-up study (also reported in chapter 3), again parts of the data from one of the labs were analyzed twice but now by other analyst teams and using SlimDEVAN in both analyses. By *comparing analysts' first and second analyses* it was possible to 1) list problem report elements that proved to be important for achieving inspectability, 2) measure analysts' consistency across the two analyses, 3) explore whether the use of SlimDEVAN influenced analysts' consistency and 4) analyze causes of inconsistencies. By *comparing problem lists across analysts* it was possible to 1) measure consistency across analysts, 2) find out whether consistency across analysts was improved by the use of SlimDEVAN and 3) analyze reasons of inconsistency across analysts.

2.1 Studying inspectability

Determining 'what makes problem reports inspectable' was done by trying to match the problem lists from the lab's first and second analyses (chapter 3). The following report characteristics proved to be helpful in tracing back problems to earlier stages of the analysis or to the original observations:

- Including descriptions of a user's actual behavior in problem descriptions, e.g., descriptions of problematic user actions;
- Making references to which users encountered a problem;
- Making references to the task context or the (sub) goal that a user was trying to accomplish at the time of the problem;
- Making clear distinctions between descriptions of problematic interactions and descriptions of interactions that are interpreted as proceeding without problems;
- Including (or making references to) overviews of observed interactions, instead of only relying on problem descriptions;
- Including accounts of users' verbal utterances, as well as clear links between such accounts and interaction overviews or problem descriptions;
- Including inferences about causes of problems (or suggestions for how to resolve problems) with clear links to behavioral problem descriptions.

SlimDEVAN problem reports have these characteristics, and apart from the last characteristic DEVAN problem reports have these characteristics as well. In addition DEVAN problem reports provide insight into analysts' interpretations of user intentions through levels of abstractions in overview tables and through segmentation of interactions.

2.1.1 How do the characteristics relate to the literature on reporting problems?

Dumas, Molich and Jeffries (2004) have published a commentary on usability problem reports collected in the well-known CUE-4 study (Molich et al. 2004). They provide four categories of advice for describing usability problems, namely 1) emphasize the positive, 2) express your annoyance tactfully, 3) avoid usability jargon and 4) be as specific as you can. Their advice aims at improving effectiveness of communicating usability problems to those who need to act on them. No recommendations concerning inspectability are mentioned.

Theofanus and Quesenbery (2005) present results from a workshop on ‘the design of effective formative test reports’. Concerning the presentation of results in formative test reports they discuss the following issues: 1) Are positive findings included in the report? 2) How are findings organized and presented? 3) How are recommendations organized and presented? 4) Are severity or priority levels included? Similar to Dumas, Molich and Jeffries (2004) their advice aims at improving the effectiveness of communicating test results to others that have to act on them. Nothing is stated about inspectability of test results.

An extensive study on reporting problems is published by Capra (2006). She presents a list of ten guidelines for writing usability problem descriptions and ranks them according to their importance from a usability practitioner’s perspective. The guidelines were developed by consulting usability practitioners through two questionnaires and a card sorting task. Her third guideline in order of importance directly relates to inspectability: “*support your findings with data*”. The summary of the guideline includes the statement: “*Provide traceability of the problem to observed data*”. How to provide traceability is not specified but some of the other guidelines overlap with the report characteristics from our study, namely 1) *describe the cause of the problem* (guideline 4) and 2) *describe observed user actions* (guideline 5). The report characteristics we suggested in our study can be used for implementing Capra’s ‘traceability’ requirement.

2.2 Analysts’ consistency across two subsequent analyses

In chapter 3 consistencies of analyst teams across two subsequent analyses were studied. Table 1 summarizes the findings. Consistencies varied considerably across the analyst teams.

What's the problem?

Studies on identifying usability problems in user tests

Table 1 Consistency across subsequent analyses as reported in chapter 3. As a measure of consistency the overlap in usability problems was determined (i.e., the number of problems found in both analyses, divided by the number of problems found in any of the two analyses).

Analysts	Used data	1 st Analysis	2 nd Analysis	Within-analysts consistency
Lab A	Lab A	Lab specific	SlimDEVAN	71,9%
Lab B	Lab B	Lab specific	SlimDEVAN	42,9%
Lab C	Lab C	Lab specific	SlimDEVAN	36,6%
Student team 1	Part from lab B	SlimDEVAN	SlimDEVAN	63,3%
Lab B	Same part as team 1	Lab specific	SlimDEVAN	42,9%
Student team 2	Part from lab B	SlimDEVAN	SlimDEVAN	71,7%
Lab B	Same part as team 2	Lab specific	SlimDEVAN	36,4%

2.2.1 Does SlimDEVAN help in improving analyst consistency?

The student teams' consistencies, based on part of the data from lab B were higher than lab B's consistencies on the same data. They were also higher than the consistencies of labs B and C on their complete data sets. Does this mean that SlimDEVAN is effective in improving analyst consistency? For multiple reasons, the definitive answer cannot be given yet: 1) Lab A's within-analyst consistency was about as high as that of student team 1, and even higher than that of student team 2, 2) results were confounded because not only the methods, but also the analysts were different (student teams vs. lab B), and 3) the higher levels of consistency for the student teams may also have been caused just by the fact that in the first and second analyses the methods were the same; in other words the choice of methods may not have mattered at all.

Future work An interesting question for follow-up research would be: "To what extent can within-analyst consistency be improved with methods like (Slim)DEVAN? To what extent is it method-independent?"

2.2.2 Analyst inconsistencies in more detail

In chapter 3, the following causes of within-analyst inconsistencies were identified:

- 1 *differences in defining 'usability problems' (12 out of 35 differences)*: differences in what is required for calling something a usability problem (e.g., defining the fact that the user needed a hint as a moment of difficulty or not, distinction between problems of usability or other problems like software bugs).
- 2 *threshold differences (8/35 differences)*: differences in an analyst's decision of how problematic an interaction should be before identifying it as a difficulty;
- 3 *false-positives (10/35 differences)*: usability problems were reported but should not have been reported, because from the detailed analysis it has become clear that it is extremely unlikely that they actually occurred (and no other proof of existence was found other than the final problem description);

- 4 *slips in the analysis leading to missed moments of difficulty (1/35 differences)*: the interaction in which the difficulty arose was observed in both analyses but in one of the analyses the difficulty was not recorded (for unknown reasons), even though it clearly was a difficulty;
- 5 *slips in the analysis leading to different problem descriptions (4/35 differences)*: missed observations or utterances leading to different interpretations of moments of difficulty and thus to different interpretations of what exactly the problems were; for example, an analyst not hearing or not noticing user's verbal utterances containing clues that help in interpreting the specific moment of difficulty.

2.2.3 How does this relate to the literature on inconsistencies in data analyses?

Hertzum and Jacobsen (2001) describe three main contributors to inconsistencies across analysts (the evaluator effect): *vague goal analyses*, *vague problem criteria* and *vague evaluation procedures*. How do these relate to our findings on within-analyst inconsistencies?

For usability tests, *vague goal analyses* relate to the tasks that test participants are asked to perform. In the conditions that we compared participants always performed the same tasks. Hence, *vague goal analyses* can not have contributed to the inconsistencies we identified. Then the question becomes 'how do the causes of inconsistencies we have identified relate to Hertzum and Jacobsen's other two *vaguenesses*':

- *Cause 1 (differences in defining 'usability problems')* obviously relates to '*vague problem criteria*'. In our study this was the largest category of differences. Most likely differences due to these causes could have been eliminated by refining our problem criteria.
- *Cause 2 (threshold differences)* can be interpreted as relating to '*vagueness in evaluation procedures*'. However, doing so implies that using more precise procedures will reduce the number of such inconsistencies. We doubt so: during the development of DEVAN we tried to get rid of such differences by trying to explore more meticulous procedures, but that proved to be ineffective.
- *Causes 3, 4 and 5 (false positives and slips)* relate to differences caused by fatigue, lack of vigilance or distraction. The combined causes 3, 4 and 5 constituted almost half of the differences. Reducing the number of such differences by applying *different*, rather than *less vague* procedures may be possible (e.g., automated logging tools reduce the chance that actions are not noticed). Complete elimination of such differences is not to be expected.

In conclusion, better (automated) tools for logging and transcribing user behavior (i.e., *different*, rather than *less vague evaluation procedures*) in combination with refined problem criteria (*less vague problem criteria*) could have improved within-analyst consistency in our case.

What's the problem?

Studies on identifying usability problems in user tests

2.3 Consistency across analysts (the evaluator effect)

In chapters 2 and 3 consistency across analysts was studied by comparing analysts' lists of problems or difficulties and determining levels of agreement. In addition, in chapter 2 differences in difficulty lists were studied in more detail to reveal what had caused the differences. Table 2 provides an overview of the levels of agreement.

Table 2 Agreement levels found in chapters 2 and 3.

Characteristics of the comparison	Agreement
1 Thermostat, 1 user, DEVAN, moments of difficulty (table 8, Ch. 2)	61%
2 TV-video, user 1, DEVAN, moments of difficulty (table 9, Ch. 2)	80%
3 TV-video, user 2, DEVAN, moments of difficulty (table 10, Ch. 2)	60%
<hr/>	
Any-two agreement of 3 labs lab A: 8 users; lab B: 6 users; lab C: 8 users	
4 Lab's own procedure, usability problems (fig. 12, Ch. 3)	30,9%
5 SlimDEVAN procedure, usability problems (fig. 12, Ch. 3)	30,9%
6 Lab's own procedure, problem category pairs (fig. 13, Ch. 3)	57,6%
7 SlimDEVAN procedure, problem category pairs (fig. 13, Ch. 3)	56,4%
<hr/>	
Agreement 8: 5 users; agreement 9: 4 users	
8 Student team 1 vs. lab B, SlimDEVAN, usability problems (fig. 15, Ch. 3)	26,8%
9 Student team 2 vs. lab B, SlimDEVAN, usability problems (fig. 15, Ch. 3)	29,4%

Before drawing conclusions from table 2 it should be noted that agreements 4, 5, 6 and 7 do not reflect 'pure' evaluator effects, because the compared problem lists originated from different test sessions. Hence, inconsistencies may have been caused either by differences in the original data or by inconsistencies across analysts. The results from the follow-up study reported in chapter 3 (agreements 8 and 9, table 2) may give a rough indication on 'how much this affected the comparisons', because for these comparisons the compared data originated from the same test sessions. In other words, differences found in comparisons 8 and 9 do reflect pure evaluator effects. Table 2 shows that agreements 4 and 5 are in the same order of magnitude as agreements 8 and 9 suggesting that agreement levels 4 and 5 were caused largely by evaluator effects, rather than by differences in the original data. Accepting this, which conclusions can be drawn from table 2?

2.3.1 Can the evaluator effect be eliminated?

The most obvious conclusion is that the evaluator effect was found in all comparisons. Even agreements 1, 2 and 3 with analysts that were extremely well-trained in the use of DEVAN (as they had developed it themselves) show evaluator effects. This suggests that the evaluator effect may not be eliminated easily (as was also suggested by Hertzum and Jacobsen (2001), Capra (2006) and others).

2.3.2 Why does the evaluator effect vary so much?

Another conclusion from table 2 is that the evaluator effect varies substantially across the nine comparisons. Explanations for this may be found in Hornbæk's (in press) discussion on the problems of matching usability problems. One of his issues concerns which entities are compared, or as Hornbæk (in press) phrases it: "*the level at which matching is done may impact evaluation results*". Agreements at the level of moments of difficulty are determined in comparisons 1, 2, and 3. Moments of difficulties are the entities at the lowest level. The next higher level of entities consists of *usability problems* as usability problems are merged moments of difficulty. Agreements at the level of usability problems are determined in comparisons 4, 5, 8 and 9. For comparisons 6 and 7 the even higher level of 'problem category pairs' were used (see chapter 3, section 4.1.1). The differences between agreements 4 and 6 as well as between 5 and 7 can be attributed to differences in entity level only. These comparisons show that consistency across evaluators is higher for the problem category pairs than for the lower level usability problems.

2.3.3 Does analyst training play a role?

Remarkably, the highest consistency across analysts is found in comparisons 1, 2 and 3, whereas in those cases 'moments of difficulty' (i.e., entities at the lowest level) were compared. This may relate to the fact that in case of agreements 1, 2 and 3 both analyst were highly trained in using DEVAN. Positive effects of training on analyst agreement were reported by Barendregt and Bekker (2006) in a study with an adaptation of the SlimDEVAN data analysis procedure.

Future research To learn more about how well DEVAN performs as a data analysis procedure in academic practice, it would be interesting to let it use by other researchers and study the effect of their training on across-analyst consistency. For example, the training on establishing DEVAN's threshold pause times could be investigated. An interesting topic for further research would be the effect of differences in threshold pause times on the detection of problems, if any.

2.3.4 Does the report format matter?

The variation in agreement levels we found in our studies also suggests that further study is needed to better understand the process of matching problems. One of the suggestions that Hornbæk and Frøkjær (2008b) make is to further study "*how the problem reporting format may matter in matching and consequently in establishing the evaluator effect*". Our data may help in suggesting answers to this question. The problem reports used for establishing agreements 4 and 6 differed from those used for agreements 5 and 7. For agreements 5 and 7 SlimDEVAN problem reports were used, and these were shown to be more inspectable than those of the lab specific analyses. Nevertheless, agreement level 4 is similar to that of 5 and agreement level 6 is similar to that of 7. This suggests that using SlimDEVAN did not affect consistency across analysts (i.e., the evaluator effect). In other words, for the evaluator effect the

What's the problem?

Studies on identifying usability problems in user tests

reporting format did not matter. On the other hand, although agreement levels 1, 2 and 3 cannot be directly compared to the others it is remarkable that these were higher than all other agreements, even though the compared entities were at a very fine level of detail.

Future research It would be interesting to further study if and to what extent structured, inspectable problem reports like those of DEVAN can improve across-analysts consistency.

2.3.5 How serious are the consequences for development contexts?

The figures on consistency of our analysts and those reported by others indicate that being consistent is extremely difficult for analysts. The question is how serious this problem is for actual product or software development practice. An important issue for that is to be aware of the fact that in many comparisons all problems are considered to be equally important. To what extent are practitioners interested in whatever usability problem will occur? Cockton (2006) states that in practice “*Only difficulties that destroy or degrade achieved value are carried forward for remediation during iteration.*” and “*What does or does not matter depends wholly on earlier translation of intended value statements into evaluation criteria*”. It would be interesting to study, whether and to what extent consistency would improve in cases where there are clear statements on what values matter to the company that asked for the test.

Future research The following topics could be considered for further research:

Is the lack of consistency within and across analysts as disastrous as suggested by the findings in chapter 2 and 3? How does the evaluator effect behave if only problems that matter are considered? Would that depend on the type of analysis procedure?

2.3.6 Inconsistencies across analysts in more detail

In chapter 2 the two analysts' lists of difficulties were compared to study causes of differences. Differences were found to relate to (see chapter 2, table 11):

- 1 *differences in detecting difficulty signal events (36-38 out of 70 differences)*: differences in assigning difficulties to inefficient task behavior, differences in interpreting vague user utterances like ‘puzzlement’, ‘confusion’ etc., differences in noticing missing actions that would eventually result in faulty execution of tasks (e.g., video recorder would not start recording at the scheduled time), differences in dealing with users setting the wrong value to some parameter, eventually resulting in faulty execution of tasks (e.g., video recorder would start recording on the wrong day).
- 2 *differences in interpreting user intentions (17-19/70 differences)*: situations in which it was unclear what the user’s intentions were and decisions on whether a user is actively trying to pursue a functional goal or is exploring the interface;
- 3 *differences in reporting (non) verbal behavior (12-14/70 differences)*: differences in reporting or not reporting nonverbal behavior (facial expressions, pointing, visually

scanning the interface, etc.), differences in decisions on whether to report evaluative remarks not directly related to actions or not (“*this thing is difficult to use*”) and differences in hearing what a participant says

- 4 *differences in (manually) logging actions (3-5/70 differences)*: e.g., analysts missing quickly correct wrong actions.
- 5 *differences in listing observed difficulties (2/70 differences)*: in some cases analysts forgot to copy an observed difficulty from the interaction table overview to the list of difficulties.

2.3.7 How does this relate to the literature on inconsistencies in data analyses?

In line with the conclusions about within-analyst inconsistencies, some of the differences could have been dealt with by using better (automated) tools for observing user behavior. This concerns items 4 and 5, which constitute only a small part of the differences (<10% of the differences).

More precise problem criteria would have helped in reducing differences related to some of the factors mentioned in item 1 and 3 (e.g., how to deal with inefficient behavior and how to deal with evaluative remarks not directly related to problematic actions). Other factors mentioned in item 1 like not noticing missing actions are hard to avoid. Finally, many of the factors in items 1, 2 and 3 are related to analysts’ beliefs, values and preferences (e.g., differences in interpreting user utterances, in inferring user intentions, differences in reporting or not reporting (non-)verbal expressions). These are persistent differences that one may try to ‘train away’ but probably this would go at the cost of biasing the analysts.

By relating our analysis of causes to the factors Hertzum and Jacobsen (2001) mentioned as contributing to the evaluator effect, we conclude that *vaguenesses in problem criteria* contributed to the evaluator effect, but that *vagueness of evaluation procedures* was not so much of a problem. Other issues not mentioned by Hertzum and Jacobsen that contribute to inconsistencies are

- the tools for observing user behavior
- the fact of e.g., not noticing missing actions because of fatigue, lack of vigilance and distraction
- differences in analysts’ beliefs, values and preferences.

Future research For dealing with differences in analysts’ beliefs, values and preferences the possibility of teams of analysts analyzing the data together could be considered. The advantage of using teams of analysts could be that multiple view points are made explicit already during the analysis and can then be discussed and possibly resolved before the final lists of problems are made.

What's the problem?

Studies on identifying usability problems in user tests

3 Applying DEVAN to a comparative study on prototypes

In chapter 4 DEVAN was applied to a comparative study on finding out how characteristics of prototypes and the prototyping situations affect test participant's behavior and resulting problem lists. Findings from two examples of prototyping techniques were compared to each other as well as to findings from a test of the functioning product of which they were a prototype. The objective of the study in the context of this thesis on the one hand was to determine whether the results of the DEVAN analysis would be inspectable enough to serve the aim of the case study and on the other hand to determine to what extent DEVAN results would be inspectable enough to successfully serve as input for a causal analysis using Zapf et al.'s error taxonomy. The approach was to use raw counts of difficulties as a vehicle to start exploring in more detail the differences in the problems and user behavior between the compared conditions.

The high level comparisons of task failures and difficulties showed only small differences between conditions. The detailed analysis of the problems showed how the users' behavior was influenced by specific prototype characteristics and by the role of the facilitator. This served as an illustration of how relevant differences in behavior and problems may level out and become invisible if only high-level summative task performance measures are used. It is important to note that although in this case the number of such differences was small, they were mainly caused by designed elements that were very specific to this exact product. It is very likely that in a product development situation these would exactly be the kind of things a developer would be interested in.

4 Conclusion

A tool for improving consistency and inspectability of user test data analysis was developed. In various studies consistency of analysts performing multiple analyses was determined as well as consistency across analysts. It was shown that much is still to be learned about the causes of such inconsistencies, its consequences and how to deal with them. Chapter 4 provided an example of how the quality of the work of analysts and comparers can be supported in studies that involve comparing usability problems. Eventually this will lead to more insight into the quality aspects of usability tests, which in turn may lead to a decrease in the number of users muttering: "*What's the problem? Why does this thing not do what I want it to do?*"

References

- Andre T.S., Hartson H.R., Belz S.M. and McCreary F.A.** (2001) The user action framework—a reliable foundation for usability engineering support tools. *Int. J. Human-Computer Studies* **54**, pp. 107-136.
- Barendregt, W. and Bekker, M.M.** (2006). Developing a coding scheme for detecting usability and fun problems in computer games for young children. *Behavior Research Methods*, **38**, 382-389.
- Barnum C.M.** 2002, *Usability Testing and Research* (Longman, New York).
- Bewley, W. L., Roberts, T. L., Schroit, D. and Verplank, W. L.** (1983). Human factors testing in the design of Xerox's 8010 'Star' office workstation. *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*. Boston, Massachusetts, United States ACM
- Blandford A.E., Hyde J.K., Green T.R.G. and Connell I.** (2008) Scoping Analytical Usability Evaluation Methods: A Case Study, *Human-Computer Interaction* **23** (3), 278-327
- Capra M. G.** (2006) Usability problem description and the evaluator effect in usability testing. PhD thesis. Virginia Polytechnic Institute and State University. Available from: http://scholar.lib.vt.edu/theses/available/etd-03222006-201913/unrestricted/MCapra_dissertation.pdf (last verified February 2009)
- Card S.K., Moran T.P. and Newell A.** 1990, The keystroke level model for user performance time with interactive systems, in J. Preece, L. Keller and H Stolk (eds) *Human-Computer Interaction: selected readings: a reader* (Prentice Hall International, Hemel Hemstead), 327-357.
- Cockton, G.,** (2004) From Quality in Use to Value in the World, *CHI 2004 Extended Abstracts*, 1287-1290.
- Cockton, G.** (2006). Focus, fit and fervour: Future factors beyond play with the interplay. *International Journal of Human-Computer Interaction*, **21**(2), 239–250.

What's the problem?

Studies on identifying usability problems in user tests

- Cockton, G. and Lavery, D.** (1999), A framework for usability problem extraction, in M.A. Sasse, C. Johnson (eds) *Proceedings of the IFIP 7th International Conference on Human-Computer Interaction — Interact'99* (IOS Press, London), 344-352.
- Cockton, G., Lavery, D., and Woolrych, A.** (2003). Inspection-based methods. In J.A. Jacko & A. Sears (Eds.), *The human-computer interaction handbook: Fundamentals, evolving technologies and emerging applications*, Mahwah, NJ: Lawrence Erlbaum Associates, Publishers. 1118-1138
- Crowe E.C. and Narayanan N.H.** 2000, Comparing interfaces based on what users watch and do, in *Proceedings of the symposium on Eye tracking research and applications 2000*, 29-36
- Daams, B. and Hariandja J.** (2006). Cross-cultural usability, testing a food processor in Indonesia and the Netherlands. *Proceedings of IEA2006: 16th World Congress on Ergonomics IEA2006*, 1-5, Amsterdam, the Netherlands: Elsevier.
- Dumas J.S., Molich R. and Jeffries R.** (2004) Describing usability problems: Are we sending the right message? *Interaction* July/August, 24-29.
- Furniss, D.** (2008). *Beyond Problem Identification: Valuing methods in a 'system of usability practice'*. PhD Thesis. University College London. Available from: <http://www.cs.ucl.ac.uk/staff/D.Furniss/Documents/08.10.28 DFurniss PhD Thesis Final copy.pdf> (last verified February 2009)
- Gould, J** (1988) "How to Design Usable Systems" in M. Helander (Ed.) *Handbook of Human-Computer Interaction*, 1st Edition, North-Holland, 757-789.
- Gould, J. and Lewis, C.** (1983). Designing for usability - key principles and what designers think. *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*. Boston, Massachusetts, United States ACM
- Gould, J. and Lewis, C.** (1985) "Designing for usability: Key principles and what designers think," *Communications of the ACM*, 28(3), 300-311.
- Gray, W. and Salzman, M.** (1998). Damaged merchandise? A review of experiments that compare usability evaluation methods, *Human-Computer Interaction* 13 (3), 203-261
- Guba E.G. and Lincoln Y.S.,** (1989). *Fourth generation evaluation*. London: Sage.
- Hacker W.,** (1986) *Arbeitspsychology [Work psychology]*. Bern: Huber.
- Hartson, H.R.** (2003). Cognitive, physical, sensory and functional affordances in interaction design. *Behaviour & Information Technology*, 22 (5), 315-338.
- Hartson, H.R., Andre, T.S. and Williges, R.C.** (2001). Criteria for evaluating usability evaluation methods. *International Journal of Human-Computer Interaction* 13 (4), 373-410.
- Hertzum, M. and Jacobsen N.E.** (2001). The evaluator effect: A chilling fact about usability evaluation methods. *International Journal of Human-Computer Interaction*, 13(4), 421-443.
- Hertzum, M., Jacobsen, N.E. and Molich, R.** (2002). Usability inspections by groups of specialists: Perceived agreement in spite of disparate observations. *Extended abstracts of the ACM CHI 2002 Conference*, 662-663, New York: ACM.

Hoenderdos, R., Vermeeren, A.P.O.S., Bekker, M.M. and Pierik A. (2002) Design for experience: the "Look, mama!" experience. *Proceedings of Interaction Design and Children Conference 2002*, 4-10, Maastricht, the Netherlands: Shaker Publishing BV.

Hollnagel E. 1993, The phenotype of erroneous action, *International Journal of Man-Machine Studies*, **39**, 1-32.

Hornbæk K. (in press) Dogmas in the Assessment of Usability Evaluation Methods. *Behaviour & Information Technology*.

Hornbæk, K. and Frøkjær, E. (2008a) A Study of the Evaluator Effect in Usability Testing, *Human-Computer Interaction*, **23** (3), 251-277

Hornbæk, K. and Frøkjær, E. (2008b) Comparison of techniques for matching of usability problem descriptions, *Interacting with Computers* **20**, 505-514

Houde, S. and Hill. C. (1997). "What Do Prototypes Prototype?" In: M. Helander, T.K. Landauer, and P.V. Prabhu, eds., *Handbook of Human Computer Interaction*, Amsterdam: Elsevier Science, pp. 367 – 381

Howarth J., Andre T.S. and Hartson R. (2007) A Structured Process for Transforming Usability Data into Usability Information. *International Journal of Usability Studies*, **3** (1), 7-23.

ISO 13407 (1999) Human-centred design processes for interactive systems. ISO.

ISO 9241-11 (1998) Ergonomic requirements for office work with visual display terminals (VDTs) Part 11: Guidance on Usability. ISO.

Jacobsen N.E. (1999), *Usability Evaluation Methods – The reliability and Usage of Cognitive Walkthrough and Usability Test*. PhD. Thesis, Copenhagen, Denmark: University of Copenhagen, Department of Psychology.

Jacobsen, N.E., Hertzum M. and John B.E. (1998), The evaluator effect in usability tests, in *Proceedings of the conference on CHI'98, Summary: Conference on Human Factors in Computing Systems* (Los Angeles, April 18-23) (ACM, New York), 255-256.

Jacobson I. (1992), *Object-Oriented Software Engineering: A Use Case Driven Approach*, Reading, Massachusetts: Addison Wesley

Jeffries, R., Miller, J. R., Wharton, C. and Uyeda, K. (1991). User interface evaluation in the real world: a comparison of four techniques. *Proceedings of the SIGCHI conference on Human factors in computing systems: Reaching through technology*. New Orleans, Louisiana, United States ACM

Jeffries, R. and Desurvire, H. (1992). Usability Testing vs. Heuristic Evaluation: Was there a Contest? *SIGCHI Bulletin*, **24** (4), 39-41

Jordan, P.W. (2000), *Designing pleasurable products: an introduction to the new human factors*, (Taylor & Francis, London).

Kanis, H. (1993). Reliability in ergonomics/human factors. *Contemporary Ergonomics*, 91-96, London: Taylor and Francis.

Kasanen, E., Lukka K. and Siitonen A. (1993). The constructive approach in management accounting. *Journal of Management Accounting Research* (5): 243-264

What's the problem?

Studies on identifying usability problems in user tests

Kemp J.A.M. and van Gelderen T. (1996), Co-discovery exploration: An Informal method for the iterative design of consumer products, in P.W. Jordan, B. Thomas, B.A. Weerdmeester and I.L. McClelland (eds) *Usability Evaluation in Industry*, (Taylor & Francis, London), 147-156.

Lavery, D. and Cockton, G. (1997). Representing predicted and actual usability problems. *Proceedings of the International Workshop on Representations in Interactive Software Development*. London, United Kingdom Queen Mary Westfield College 97-108

Lavery, D., Cockton, G. and Atkinson M. (1997). Comparison of evaluation methods using structured usability reports. *Behaviour & Information Technology*, 16 (4), 246-266.

Law E. L.-C. and Hvannberg E.T. (2004). Analysis of combinatorial user effect in international usability tests. *Proceedings of ACM CHI 2004 Conference on Human Factors in Computing Systems*. New York: ACM.

Lewis, J. R. (1994). Sample sizes for usability studies: Additional considerations. *Human Factors*, 36, 368-378

Lindgaard G. and Chattrachart J. (2007) "Usability testing: What have we overlooked?", *Proceedings of the SIGCHI conference on Human factors in computing systems, CHI'07*, ACM, April, San Jose, U.S.A., pp. 1415 - 1424.

Mayhew, D. J. (1999). *The usability engineering lifecycle: A practitioner's handbook for user interface design*. San Francisco, California: Morgan Kaufmann Publishers.

Molich R., Bevan N., Curson I, Butler S., Kindlund E., Miller D. and Kirakowski J. (1998), Comparative evaluation of usability tests, in *Proceedings of the Usability Professionals Association 1998 (UPA98) Conference, Washington D.C. USA, June 25-26, 1998*.

Molich R., Damgaard Thomsen A., Schmidt L., Ede M., Oel W. Van and Arcuri M (1999), Comparative evaluation of usability tests, in *Proceedings of CHI'99, extended abstract*, 83-84. Data available at <http://www.dialogdesign.dk/cue.html> (last verified February 2009)

Molich, R., Ede M.R., Kaasgaard K. and Karyukin B. (2004). Comparative usability evaluation. *Behaviour & Information Technology*, 23 (1), 65-74.

Molich R. and Dumas J.S. (2008), "Comparative Usability Evaluation (CUE-4)," *Behaviour & Information Technology*, 27 (3), 263-281

Nardi B.A. 1996, Activity theory and human-computer interaction, in B.A. Nardi (ed.) *Context and Consciousness: Activity theory and human-computer interaction* (MIT Press, Cambridge MA), 7-16.

Newell A. 1990, *Unified Theories of Cognition* (Harvard University Press, Cambridge MA).

Newman W.M. 1998, Commentary on 'Damaged Merchandise?' *Human-computer interaction*, Vol. 13, No. 3 (Special issue on 'Experimental Comparisons of Usability Evaluation methods), 316 - 323.

Nielsen, C.M., Overgaard, M., Pedersen, M.B., Stage, J. and Stenild, S. (2006) It's Worth the Hassle! The Added Value of Evaluating the Usability of Mobile Systems in the Field, In Nordichi 06: *Proceedings of the Fourth Nordic Conference on Human-Computer Interaction 2006*. pp. 272-280.

References

- Nielsen, J.** (1989) Usability Engineering at a discount, in *Designing and Using Human-Computer Interface and Knowledge Based Systems*, G.Salvendy & M.J. Smith (eds.), Amsterdam: Elsevier, pp. 394-401.
- Nielsen, J.** (1997), Usability testing, in G. Salvendy (ed.), *Handbook of Human Factors and Ergonomics*, Second Edition, (John Wiley and Sons, New York), 1543-1568.
- Nielsen, J.** (2000), "Why You Only Need to Test With 5 Users", *Alertbox March 19, 2000*, at <http://www.useit.com/alertbox/20000319.html>, accessed on 5 February 2009
- Nielsen, J.** (2008). Heuristic Evaluation. Retrieved 1 August 2008, from <http://www.useit.com/papers/heuristic>
- Nielsen, J. and Landauer, T. K.** (1993). A mathematical model of the finding of usability problems. *Proceedings of the INTERACT '93 and CHI '93 conference on Human factors in computing systems*. Amsterdam, The Netherlands ACM 206-213
- Nielsen, J. and Molich, R.** (1990). Heuristic evaluation of user interfaces. *Proceedings of the SIGCHI conference on human factors in computing systems: Empowering people*. Seattle, Washington, United States ACM
- Noiwan J. and Norcio A.F.** (2006). Cultural differences on attention and perceived usability: Investigating color combinations of animated graphics. *International Journal of Human-Computer Studies* 64 (2), 103-122.
- Noldus** (2002), *The Observer Video-Pro*. <http://www.noldus.com/products/index.html>
- Norman D.A.** 1986, Cognitive engineering, in D. A. Norman and S. Draper (eds.) *User centred system design: New Perspectives on human-computer interaction* (Lawrence Erlbaum Associates, New Jersey), 31-62.
- Oosterholt R., Kusano M. and de Vries G.** 1996 Interaction design and human factors support in the development of a personal communicator for children, in *Conference proceedings on Human factors in computing systems*.
- Öztoprak, A. and Erbug, C.,** (2006) Field versus laboratory usability testing: A first Comparison. *Developments in Human Factors in Transportation, Design, and Evaluation*", pp. 205-212.
- Park K.S. and Lim C.H.** (1999). A structured methodology for comparative evaluation of user interface designs using usability criteria and measures. *International Journal of Industrial Ergonomics* 23, 379 - 389.
- Polson, P. G., Lewis, C., Rieman, J. and Wharton, C.** (1992). Cognitive Walkthroughs - a Method for Theory-Based Evaluation of User Interfaces. *International Journal of Man-Machine Studies*, 36 (5) 741-773
- Preece J., Rogers Y and Sharp H.** 2002, *Interaction design: beyond human-computer interaction* (John Wiley and Sons, New York).
- Rasmussen J.** (1982). Human errors: A taxonomy for describing human malfunction in industrial installations. *Journal of Occupational Accidents*, 4, 311-335.
- Rooden M.J. and Kanis H.** 2000, Anticipation of usability problems by practitioners, in *Proceedings of the Human Factors and Ergonomics Society 44th Annual Meeting (San Diego)* 6-941 – 6-944.

What's the problem?

Studies on identifying usability problems in user tests

- Rowe, A.L., Lowry, T., Halgren S.L. and Cooke N.J.** (1994), A comparison of usability evaluations conducted by different teams, in *Proceedings of the CHI'94 Conference Companion on Human Factors in Computing Systems (Boston, April 24-28)* (ACM, New York), 109-110.
- Sauer J., Franke H. and Ruettinger B.**, (2008) Designing interactive consumer products: Utility of paper prototypes and effectiveness of enhanced control labeling. *Applied Ergonomics* 39, 71-85.
- Siegel, S. and Castellan, N.J.** 1988, *Nonparametric statistics for the behavioural sciences* (McGraw-Hill Book Company, New York), 144-151.
- Spool, J. and Schroeder, W.** (2001). Testing web sites: Five users is nowhere near enough. *Extended abstracts of CHI 2001*, 285-286.
- Steen M.G.D.** (2008) *The fragility of human-centred design*. PhD thesis (28 November 2008), Delft University of Technology, Industrial Design Engineering.
- Streefkerk, J.W., van Esch-Bussemakers, M.P., Neerinx, M.A., and Looije, R.** (2008). Evaluating Context-Aware Mobile User Interfaces. In: J. Lumsden (Ed.), *Handbook of Research on User Interface Design and Evaluation for Mobile Technology*. Chapter XLV (pages 759-779). Hershey PA, USA: IGI Global.
- Sutcliffe A., Ryan M., Doubleday A. and Springett M.** (2000), Model mismatch analysis: towards a deeper explanation of user's usability problems, *Behaviour and Information Technology*, 19 (1), 43-55
- Theofanus M. and Quesenbery W.** (2005) Towards the Design of Effective Formative Test Reports. *Journal of usability studies* 1 (1), 27-45.
- Vermeeren, A.P.O.S.**, (1999). Designing scenarios and tasks for user trials for home electronic devices. In W.S. Green & P.W. Jordan (Eds.), *Human Factors in Product Design: current practice and future trends* (pp 47-55). London: Taylor & Francis, 47-55.
- Vermeeren, A.P.O.S.** (2003). *The DEVAN checklist for detecting problems in interactions: user manual (checklist version 1.1)* (Technical Report). Delft, The Netherlands: Delft University of Technology, Faculty of Industrial Design Engineering.
- Vermeeren, A.P.O.S.** (2004). Structured video analysis of user test data, using the DEVAN tool. In Ç. Erbuğ (Ed.) *Usability testing: Methods, experiences, achievements* (pp. 123-140). Ankara, Turkey: Middle East Technical University, Faculty of Architecture.
- Vermeeren, A.P.O.S., den Bouwmeester, K., Aasman, J., and de Ridder, H.** (2002). DEVAN: A tool for detailed video analysis of user test data. *Behaviour & Information Technology*, 21 (6), 403-423 (chapter 2 of this thesis).
- Vermeeren, A.P.O.S., van Kesteren, I.E.H., and Bekker M.M.** (2003). Managing the Evaluator Effect in User Testing. *Proceedings of INTERACT'03*, Amsterdam, The Netherlands: IOS Press, 647-654.
- Vermeeren A.P.O.S., Koenderink-van Doorn A.J. and Ridder H. de** (2006) Reliability of a checklist-based user test data analysis procedure. *Proceedings IEA 2006 Congress: Meeting Diversity in Ergonomics*, Amsterdam, The Netherlands: Elsevier.
- Vermeeren, A.P.O.S., Bekker, M.M., van Kesteren, I.E.H. and de Ridder, H.**, (2007). Experiences with structured interviewing of children during usability tests. In L.J. Ball et

References

al. (eds.) *Proceedings of HCI 2007, The 21st British HCI Group Annual Conference*, 139-146. Swindon, UK: BCS.

Vermeeren A.P.O.S., Attema J., Akar E., Ridder H. de, Doorn A.J. van, Erbug C., Berkman A.E. and Maguire M.C. (2008) Usability Problem Reports for Comparative Studies: Consistency and Inspectability. *Human-Computer Interaction* 23 (4), 329-380 (chapter 3 of this thesis)

Vermeeren, A.P.O.S., Kort J., Cremers, A.H.M. and Fokker J., (2008) Comparing UX Measurements, a case study. *5th COST294-MAUSE Open Workshop on Valid Useful User Experience Measurement (VUUM)*. Reykjavik, Iceland.

Virzi R.A. (1989) What can you learn from a low-fidelity prototype, in *Proceedings of the Human Factors Society 33rd Annual Meeting*, Denver, Colorado, 1989, pp.224-228.

Virzi R.A. (1992). Refining the test phase of usability evaluation: how many subjects is enough? *Human Factors* 34 (4), 457 – 468.

Watson, B. (1968) *The Complete Works of Chuang Tzu*, Columbia University Press, Ch 26, p 302.

Zapf, D., Brodbeck, F. C, Frese, M., Peters M. and Prümper, J. (1992) Errors in working with office computers: a first validation of a taxonomy for observed errors in a field setting, *International Journal of Human-Computer Interaction*, 4, 311-339.

What's the problem?

Studies on identifying usability problems in user tests

Appendix A

Examples from the lab specific analyses

In the initial analyses that the three teams performed, each team used the procedures as they always did in their regular professional practice. Team A manually logged their sessions, with problem descriptions embedded in the loggings (see Figure A1 for an example). Team B logged their session using dedicated software (see Figure A2 for an example log) and then wrote a separate report summarizing their findings and suggestions for improvement (see Figure A3). Team C made hand-written notes of their sessions and then reported their findings in a separate document (see Figure A4 for an example of how they reported their findings).

What's the problem?

Studies on identifying usability problems in user tests

<Name> 41		High	Female		
Task	Effectiveness	Problem	Suggestion	Comments	Meets expectations?
Stop	1	Pressed rotary knob, turned. Pressed 'stop' but puzzled when she saw 'menu' and 'cooking'. Pressed 'stop' unintentionally.	Display may turn dark; feedback or a text may appear on the screen	Not safe, Didn't provide feedback	

Figure A1 Example of part of a session log from team A (lab specific analysis). Task: stop cooking. In the top row, characteristics of the participant are given. The various columns show (1) the task, (2) task effectiveness (either 1 or 0), (3) free-form problem description/logged actions, (4) evaluator's suggestions for solutions to the problem, (5) general comments, (6) answer to the question whether the product met the participant's expectations with respect to this task.

	[SYS][switched to task Prolong the cooking time]	36	Prolong the cooking time	1589	
28-10-2003 10:35	start	36	Prolong the cooking time	1591	NL281003_930_riette.avi
28-10-2003 10:35	I will make it thirt minutes ...iot his right	36	Prolong the cooking time	1593	NL281003_930_riette.avi
28-10-2003 10:35	I forgot to check how many minutes you still have to go...what is twenty minutes more	36	Prolong the cooking time	1602	NL281003_930_riette.avi
28-10-2003 10:36	...Im not sure I forgot to check how many minuytes to go...	36	Prolong the cooking time	1624	NL281003_930_riette.avi
28-10-2003 10:36	stops and explains	36	Prolong the cooking time	1650	NL281003_930_riette.avi
28-10-2003 10:36	you did not check the temperaty	36	Prolong the cooking time	1670	NL281003_930_riette.avi
28-10-2003 10:36	I will try	36	Prolong the cooking time	1676	NL281003_930_riette.avi
28-10-2003 10:36	goes to edit	36	Prolong the cooking time	1679	NL281003_930_riette.avi
28-10-2003 10:37	goes to temperatue	36	Prolong the cooking time	1683	NL281003_930_riette.avi
28-10-2003 10:37	change temperature	36	Prolong the cooking time	1687	NL281003_930_riette.avi

Figure A2 Example of a part of team B's session logs (lab specific analysis). Columns show (from left to right): date and time; recorded events; code for participant; task; video frame number; reference to video file.

Task 1 Stop the oven

Problems related to task

Find the stop button:

Users find the stop-button easily, press the button, but are then confused by the feedback the oven provides.

Reasons:

1. Display still says "cooking", indicating a menu-item. Novice users interpret it as feedback: the oven is still cooking.
2. When users go up one level, they find an icon that shows a cooking pan. It is animated and is interpreted by novice users as: the oven is still cooking
3. There is no tactile, visual or audible feedback that indicates that the oven stopped cooking after pressing the stop button.

Eventno	Eventlog	Msec.	Video file
794	Did things...I pushed a button...I think the oven stopped	4819	NL281003_930_riette.avi
375	Goes to settings	5156	NL281003_1400_robort.avi
653	Does not see that it has stopped already	5482	NL30102003_Marion.avi
320	Is it stopped...goes to start now	6025	NL031103_1400_Femke.avi
518	What do you think when you saw this...what should I do now	6244	NL04112003_900_karin_1.avi
481	Tries to turn	6749	NL051103_1400_wim.avi

Weight:

Weight = high. In emergency situations novice users will not be sure whether the oven stopped and especially in dangerous situations they may try to stop it again or to verify whether it stopped.

Suggestions:

1. Give feedback when stopping the oven, for example a small screen indicating that the oven stopped. Or an indicator of the on/off status of the oven that can be seen in each menu. Suppose that the oven evolves and users can change settings without stopping cooking, such an indicator maybe very useful (see findings on alarm-clock)
2. The word "cooking" is very active: it seems to indicate the status of the oven. Maybe there is an alternative word. After selecting suggestion 1 the confusion is probably already over.
3. Some users preferred a stop/start button. This is also a way of providing immediate feedback: a clear start/stop button that is in or out depending on the status of the oven.

Figure A3 Example of how team B reported its findings (lab specific analysis).

What's the problem?

Studies on identifying usability problems in user tests

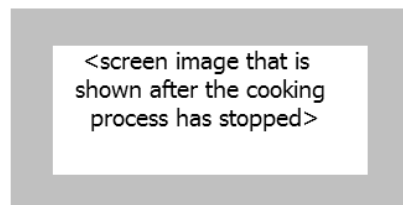
4. Problems and issues with control panel listed by task

This section describes the main problems identified during observation of the sample users. Suggestions for change are given in the tables below. *More detailed recommendations for change listed per screen are shown in Appendix 1.*

Scenario 1: Stop the cooking

Problem/comment	Recommendation
There is no clear feedback after pressing 'stop', the cooking process has stopped. 7 users continued interacting thinking that oven was still cooking. One user tried to reduce the temperature while another thought oven temperature would show it cooling.	After pressing stop, user should go to a screen that indicates stop and gives the options to 'stop completely' or 'continue' (default – stop completely). After stop, system should return to main menu.
When user gets back to main menu, the current option is 'cooking', which indicates that the oven is still cooking.	Maybe re-label the tab as 'cook' or 'cook setup'
Left hand arrow indicating that there is a submenu available is not clear.	Consider another way to indicate lower level menu, e.g. place arrow after the menu option, or place 3 dots after the menu option.

(from the report's appendix 1:)



- After the user presses Stop, there is no feedback that the oven has stopped cooking. Pressing Stop should lead to a screen with a heading and two options:

COOKING PAUSED
Stop completely
Continue

This would give the user the option to either stop completely, or to check the food and continue. If the user presses 'Stop completely' then the cooker should return to the Main Menu.

Figure A4 Examples from team C's report (lab specific analysis).

Appendix B

Examples from the SlimDEVAN analyses

In the reference analyses, teams were asked to use SlimDEVAN. Although, to some extent, SlimDEVAN prescribes how to perform the analysis, the teams' reports showed differences in their implementations of it. Figures B1, B3 and B5 show how the teams (A, B and C, respectively) logged their sessions. Figures B2, B4 and B6 show how the teams (A, B and C, respectively) reported the usability problems they had identified.

What's the problem?

Studies on identifying usability problems in user tests

USER 7: FEMALE, 41, HIGH INCOME		
Actions (time stamp)	Verbal utterances, user behaviour	Breakdown signal types
(00:27)		<u>TASK: Stop the oven cooking</u>
presses rotary knob (00:28)	Shall I stop now?	ACT
turns rotary knob (00:29)		ACT
Repeated turning and pressing rotary knob <0:29: 00:32> (00:33)	Didn't stop!	REP, ACT RAND and DSF RAND: After the end of the task it was indicated by the user that she hadn't seen stop button, she had pressed it unintentionally. DSF: Surprized by the action's effect, can't understand why she couldn't stop
TASK GOAL ACHIEVED (00:34)		DSF Doesn't understand action's effect
(00:35)	Himm...	DSF Surprised by the action's effect
cooking diplay appears (00:43)	Now select something ?....	INTN Specifies an action that she thinks is needed
(00:44)	What shall I do?	PUZZ Doesn't know what action is needed to stop
(00:53)	Didn't stop !	WEX Oven has already stopped but she formulates a wrong explanation for the display

Figure B1 Example of a log made by team A (SlimDEVAN analysis).

Appendix B
Examples from the SlimDEVAN analyses

<Name>		
Time stamp and signal codes	Free-form difficulty description	Inferences about what design elements may have caused the difficulties to occur.
(00:28) ACT	User presses rotary knob to <i>stop</i> the oven	The rotary knob is the most dominant element among the controls, so that user is directed to that without much intention. Furthermore, <i>stop</i> is not sufficiently expressed.
<0:29: 00:32> REP, ACT	User rotates and presses the knob several times to <i>stop</i> .	User expects to control the oven with the knob, just as she uses a conventional oven. This is a problem regarding user's expectation.
(00:53) WEX	Although she successfully stopped the oven she expresses that she was not successful.	After the oven is stopped user is not informed. The info present in the following screen actually misleads the user.

Figure B2 Example of usability problem list as reported by team A (SlimDEVAN analysis).

	[SYS] [switched to task Stop the cooking]	37	Stop the cooking	289	
28-10-2003 14:15	...go ahead	37	Stop the cooking	344	NL281003_1400_robort.avi
28-10-2003 14:15	this is the oven	37	Stop the cooking	349	NL281003_1400_robort.avi
28-10-2003 14:15	presses...loooks, turns the button	37	Stop the cooking	352	NL281003_1400_robort.avi
	[DEVAN] [ACT], Wrong Action: presses...loooks, turns the button	37	Stop the cooking	357	NL281003_1400_robort.avi
28-10-2003 14:15	presses at back	37	Stop the cooking	360	NL281003_1400_robort.avi
28-10-2003 14:15	goes to menu	37	Stop the cooking	363	NL281003_1400_robort.avi
28-10-2003 14:15	recipes	37	Stop the cooking	369	NL281003_1400_robort.avi
	[DEVAN] [ACT], Wrong Action: recipes	37	Stop the cooking	369	NL281003_1400_robort.avi
28-10-2003 14:15	goes to settings	37	Stop the cooking	375	NL281003_1400_robort.avi
28-10-2003 14:15	alarm	37	Stop the cooking	378	NL281003_1400_robort.avi
28-10-2003 14:15	status of alarm	37	Stop the cooking	380	NL281003_1400_robort.avi
28-10-2003 14:15	cooking	37	Stop the cooking	383	NL281003_1400_robort.avi
28-10-2003 14:15	turns knob	37	Stop the cooking	389	NL281003_1400_robort.avi
28-10-2003 14:15	I'm looking for a stopbutton	37	Stop the cooking	394	NL281003_1400_robort.avi
	[DEVAN] [SEARCH], Searches for Function: Im looking for a stopbutton	37	Stop the cooking	394	NL281003_1400_robort.avi
28-10-2003 14:16	I can't find it	37	Stop the cooking	402	NL281003_1400_robort.avi

Figure B3 Example of part of a session log made by team B (SlimDEVAN analysis). Columns (from left to right): time and date; logged events; code indicating participant; task; video frame number; reference to video file.

What's the problem?

Studies on identifying usability problems in user tests

Usability issues related to the cooking screens.

MAIN

1. When alarm is set it is not clear if time indicates time until alarm or indicates the actual time

STARTING/STOPPING

1. All users press the stop button when asked to stop the cooking process. However: the feedback is not clear. The "cooking" title above the menu, the animated and moving "Cooking"-picture suggest that the oven is still operating.

2. Cooking is interpreted as "Boiling". The animated gif with the boiling pan supports this interpretation. It is better to have a picture of an oven.

Weight

The issues related to starting and stopping the oven are found several times in each session (in total 34 times).

The issues related to editing were found 16 times, and in each session. The issues related to the menu, status were found in at least three sessions. The defrost issues were found in two sessions.

The start/stop issues are therefore probably quite general for the user population, also the issues related to the edit-menu. The issues related to status, menu and defrost were found in specific sessions in which users were less experienced.

Validity

Start and stop issues and issues related to the edit cooking are quite central to the operation of the oven. In general it can be concluded that users can learn to operate the oven, without a manual but they need some learning time. Reaching errorless and routine performance will probably take some time, for the less used functions it may be difficult to reach.

Suggestions

1. If applicable, present in the opening screen also information on selected program and alarm settings.

2. The term "Cooking" is confusing. Although it finally remains the only option for setting settings after users have tried the other items (setting duration in the time dialog or in the alarm dialog, looking in recipes). The icon is associated with boiling (potatoes, vegetables) and not with an oven.

The cooking icon is animated, suggesting that the oven is actually working. Also the title "cooking" suggests that the oven is working. A clear indication of the status of the oven, a program or alarm could help users to understand what the oven is doing when they are in the main menu.

3. Pressing the stop-button should give feedback, for instance a sound.

FINDING	641	D_cooking_adaptation of parameters: When changing duration the old temperature is not shown. If user forgets: cannot cancel changes. Not clear if duration indicates minutes to go or total duration
----------------	-----	------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

[DEVAN][COMM], Comment: ... He is on bottom now... it works... the minutes I changed... I suppose it is clear... is not sure if minutes were actually changed (program does not take change in time !!!)

session:	39	Pre-Interview	video	1259
-----------------	----	---------------	--------------	------

[DEVAN][COMM], Comment: was not clear that minutes were added... I was not sure if duration was the total time or the time left

session:	41	Prolong the cooking time	video	1280
-----------------	----	--------------------------	--------------	------

[DEVAN][DIFF], Execution Difficulty: ... Im not sure I forgot to check how many minuytes to go...

session:	36	Explain the display	video	1625
-----------------	----	---------------------	--------------	------

Figure B4 Parts from the usability problem report of team B (SlimDEVAN analysis).

TEBA OVEN STUDY	SUBJECT 5: <NAME>	
Actions/Times (minutes and seconds)	Verbal utter- ances/User behaviour	Difficulty signals
0.11 Pressed STOP.	"The natural thing would be to press the button that says Stop. But I don't think that's right. Perhaps I'll twiddle that knob."	Task 1: Stop the cooker User achieves task but does not realise it.
0.17 Tries out knob and moves up to MODE.	"But that doesn't do anything."	ACT
0.30 Down to DURATION and up to MODE.	"So..."	ACT
0.32 Selects MODE		ACT
0.36 Twiddles knob within mode options.		ACT
0.39 Presses BACK to go to MODE	"The oven's going to blow up I suppose. I'm stuck. I don't know what to do. I've probably messed it up."	PUZZ
0.44 Task completed		

Figure B5 Example from a log file made by team C (SlimDEVAN analysis).

<p><i>Problem list following by subjects who experienced them (numbered 1 to 8).</i></p> <p>Lack of feedback when stopping oven (1,2,3,4,5,6,7,8)</p> <p>Fixed order of making settings in cooking menu (1,2,3,4,5,6,7)</p> <p>Hesitating or not realising that they should select COOK to start cooking (1,2,3,4,5,6,7,8)</p> <p>Expects selection bar in View and Cook screen (7)</p> <p>Did not realize had to press VIEW AND COOK (4)</p>

Figure B6 Example from the problem list made by team C. (SlimDEVAN analysis).

What's the problem?

Studies on identifying usability problems in user tests

Summary

“*What’s the problem? Why does this thing not do what I want it to do?*” – this is the kind of muttering one may hear when someone is trying to use a newly bought product. The user of the product has just encountered a difficulty in use, a usability problem. During the process of developing products it is hard to predict such problems. This especially holds for interactive products with embedded software. In user tests conducted during the design process analysts try to foresee which problems people will run into when using a product. Once they have identified and understood the problems, product developers may attempt to redesign the product so that the risk of users encountering usability problems will be minimized.

In this thesis we deal with methodological issues of usability (or user) tests, in particular with issues on how analysts extract lists of usability problems from observed user behavior. Doing this in a consistent manner has proven to be very difficult. Two types of consistency turn out to be relevant. The first type called *within-analyst consistency* concerns issues like ‘were all observations analyzed in the same way?’ and ‘if the analyst would do the analysis again would (s)he find the same usability problems?’ The second type of consistency deals with the following question: ‘do multiple analysts find the same problems?’ This is called *across-analysts consistency*. The finding that multiple analysts hardly ever identify the same set of usability problems is called *the evaluator effect*. The term evaluator is often used as a more general term for analyst.

Sources of inconsistencies can be found in almost all activities of a data analysis process. Some of these concern the fact that analysts may become tired, less attentive or distracted during the analysis. The number of such inconsistencies can be reduced by using appropriate procedures and tools. Important sources of across-analyst inconsistencies are differences in analysts’ beliefs, values or preferences. Such inconsistencies should not be seen as errors in the analysis but as reflections of differences in the interpretation of user behavior. Trying to align these interpretations is not always desir-

What's the problem?

Studies on identifying usability problems in user tests

able; instead they should be exposed by making the analysis *inspectable* so that interpretations can be discussed with others. *Inspectability* means that insight is provided into all procedures, ranging from observing and recording user behavior to identifying and describing usability problems, so that it can be made clear 'which observations led to the identification of which usability problem'.

The main objectives of this thesis are to study issues of consistency in identifying usability problems in user tests and to devise a way to deal with inconsistencies by

- 1 developing a procedure that makes the analysis documentable and inspectable,
- 2 studying to what extent the developed procedure exposes possible sources of inconsistency and manages to reduce inconsistencies caused by fatigue, lack of vigilance and distraction,
- 3 applying the developed procedure to a comparative study in which the effect of using prototypes (instead of functioning products) in user tests is studied in detail.

1 Developing the (Slim)DEVAN data analysis procedure

Chapter 2 presents the DEVAN (DEtailed Video ANalysis) procedure for making user test data analyses documentable and inspectable. In DEVAN tables representing interactions at various levels of abstraction are created to make an analysis inspectable by allowing results from the various steps in the analysis to refer back to data elements in these tables. Consistency in creating the overview tables is facilitated by segmenting interactions on the basis of threshold times for pauses between actions. A checklist of observable signal events helps analysts to remain consistent in detecting when a usability problem occurs. Difficulties in interactions are described by means of a standard format containing references to specific elements in the interaction overview tables. In chapter 3 the development of an adaptation of DEVAN named SlimDEVAN is reported. SlimDEVAN was developed as a simpler, less time-consuming version of DEVAN. To this end, the multiple levels of abstractions were removed from the interaction overview tables and no prescribed way of segmenting interactions was included. The checklist of signal events can be used in the same way as in DEVAN and now problem reports also provide room for reporting possible causes of difficulties. The report formats are simpler than in DEVAN without losing the necessary references to elements of the interaction overviews.

2 Studying inspectability and consistency

Chapters 2 and 3 report studies in which (Slim)DEVAN was used to investigate various issues of inspectability and consistency. Chapter 2 describes a study in which DEVAN's inspectability was tested by comparing multiple lists of interaction difficulties extracted from the same videotaped test sessions. The lists were created by two analysts (the developers of DEVAN) working independently of each other. Chapter 3 presents studies on inspectability and consistency in which problem reports from three professional usability labs were compared. The three labs had tested the same product using a test protocol they had developed collaboratively. Each lab had analyzed their data twice: first by using their own regular data analysis procedure and subsequently (a few weeks

later) by applying SlimDEVAN. In a follow-up study (also reported in chapter 3), again parts of the data from one of the labs were analyzed twice but now by other analysts and using SlimDEVAN in both analyses.

Within-analyst consistency and inspectability Analysts' first and second analyses were compared to identify problem report elements important for achieving inspectability, to measure analysts' consistency across two analyses and to analyze causes of inconsistencies. We identified a list of problem report characteristics that helps in making them inspectable. Within-analyst consistency was found to vary considerably. An analysis of the causes of within-analyst inconsistencies revealed that advanced (automated) tools for logging and transcribing user behavior in combination with refined problem criteria may improve within-analyst consistency. Comparison of the within-analyst consistency measures tends to show higher within-analyst consistency when SlimDEVAN was used in both analyses. Further research is needed to study to what extent this can be attributed to the use of SlimDEVAN.

Across-analysts consistency and inspectability Usability problem lists and lists of difficulties were compared across analysts to measure the evaluator effect, to find out whether consistency across analysts can be improved by the use of (Slim)DEVAN and to analyze sources of inconsistencies across analysts. The evaluator effect was found to vary considerably across all our comparisons and we had to conclude that the evaluator effect cannot be eliminated easily. Our findings suggest that the variation in the evaluator effect has been caused by differences in the units chosen for calculating this effect and by differences in the extent to which analysts were trained users of (Slim)DEVAN. Whether structured, inspectable problem reports can improve across-analysts consistency remained unclear. The study reported in chapter 2 revealed that vaguenesses in problem criteria contribute to the evaluator effect. Other causes of across-analyst inconsistencies were differences in analysts' beliefs, values and preferences, the tools for observing user behavior and the fact that missing actions were not noticed because of fatigue, lack of vigilance and distraction.

We conclude that it is interesting to study whether and to what extent consistency will improve when it is clear which values matter to the company that asked for the test and these values are taken into account while determining the evaluator effect. For dealing with differences in analysts' beliefs, values and preferences we propose to consider the possibility of teams of analysts analyzing the data together. The advantage of using teams of analysts may be that multiple viewpoints can be made explicit already during the analysis so that they can be discussed and possibly resolved before the final lists of problems are made.

3 Applying DEVAN to a comparative study on prototypes

In chapter 4 DEVAN is applied to a comparative study aimed at finding out how characteristics of prototypes and prototyping situations affect a test participant's behavior and the problem list resulting from the test. Findings from two examples of prototyp-

What's the problem?

Studies on identifying usability problems in user tests

ing techniques were compared to each other as well as to findings from a test with the actual product. In the context of this thesis the objective of this study was to determine whether the DEVAN analysis is inspectable enough to serve the aim of the comparative study and whether the results of a DEVAN analysis are suitable to be successfully linked with an error taxonomy for performing causal analyses.

Both objectives were met in the comparative study. Detailed problem analyses showed how the behavior of users was influenced by prototype characteristics and by the role of the facilitator. We also observed that summative task performance measures tend to level out interesting differences in user behavior and experienced difficulties. These differences appeared to be mainly caused by product elements that made this product outstanding relative to competitors' products.

4 Conclusion

This thesis has demonstrated how user test data analyses suffer from persistent inconsistencies. Making the analyses inspectable is proposed as a way of dealing with the more persistent inconsistencies while advanced (automated) observation tools and more precise problem criteria are proposed for reducing less persistent inconsistencies. We suggest that further research should focus on consequences of inconsistencies in actual product development contexts. Eventually this will lead to more insight into the quality aspects of user tests, which in turn may lead to a decrease in the number of users muttering: *"What's the problem? Why does this thing not do what I want it to do?"*

Arnold Vermeeren, March 2009

Samenvatting

“Wat is 't probleem? Waarom doet dit ding niet wat ik wil?” – dat is het soort gemopper dat je te horen kan krijgen, als iemand probeert een net gekocht product te gebruiken. De gebruiker van het product is dan op een moeilijkheid gestuit bij het gebruik, een bruikbaarheidsprobleem (‘usability problem’). Zulke problemen zijn tijdens het productontwikkelingsproces moeilijk te voorspellen. Dit geldt in het bijzonder voor interactieve producten met ingebouwde computerprogrammatuur. In gebruiksonderzoeken tijdens het ontwerpproces proberen analisten vooraf te ontdekken welke problemen mensen kunnen ondervinden bij het gebruik van het product. Als ze die problemen eenmaal gevonden en begrepen hebben, kunnen productontwikkelaars gaan proberen het ontwerp van het product zo te veranderen dat de kans op bruikbaarheidsproblemen minimaal wordt.

In dit proefschrift worden methodologische aspecten van gebruiksonderzoeken behandeld, en vooral die aspecten die te maken hebben met de wijze waarop analisten lijsten met bruikbaarheidsproblemen afleiden uit waargenomen gedrag van gebruikers. Gebleken is dat het erg moeilijk is om dit op een consistente manier te doen. Twee soorten consistentie blijken daarbij relevant te zijn. Bij de eerste soort, genaamd *binnen-analist consistentie*, gaat het om vragen als ‘zijn alle observaties op dezelfde manier geanalyseerd?’ en ‘als de analist de analyse nog eens zou uitvoeren, zou die dan weer dezelfde problemen vinden?’ Bij de tweede soort consistentie gaat het om de vraag ‘vinden verschillende analisten wel dezelfde problemen?’ Dit heet *tussen-analisten consistentie*. Het verschijnsel dat verschillende analisten zelden dezelfde set van bruikbaarheidsproblemen herkennen, wordt het *beoordelaarselect* genoemd. ‘Beoordelaar’ wordt vaak gebruikt als een meer algemene term voor analist.

Inconsistenties kunnen hun oorsprong hebben in bijna alle activiteiten binnen een data-analyseproces. Bepaalde inconsistenties ontstaan doordat analisten gedurende de analyse soms moe en minder oplettend worden of niet steeds geconcentreerd zijn. Het

Wat is 't probleem?

Onderzoek naar het herkennen van bruikbaarheidsproblemen in gebruiksonderzoeken

aantal van dit soort inconsistenties kan verkleind worden door het gebruik van geschikte procedures en hulpmiddelen. Daarnaast zijn er tussen analisten veel inconsistenties die voortkomen uit verschillen in opvattingen, waarden of voorkeuren van die analisten. Dergelijke inconsistenties kunnen beter niet gezien worden als fouten in de analyse, maar als het resultaat van verschillen in de interpretatie van het gedrag van gebruikers. Het is niet altijd wenselijk om te proberen die interpretaties gelijk te trekken. Beter is het om de verschillen in interpretatie bloot te leggen door ervoor te zorgen dat de analyse stap voor stap te volgen ('inspectable') is, waardoor het mogelijk wordt die verschillen met anderen te bespreken. Een analyse is inspecteerbaar, als inzicht wordt gegeven in alle procedures, van het observeren en registreren van het gedrag van gebruikers tot het herkennen en beschrijven van bruikbaarheidsproblemen, en als duidelijk wordt 'welke observatie heeft geleid tot welk bruikbaarheidsprobleem'.

De hoofddoelstellingen van dit proefschrift zijn het bestuderen van consistentievraagstukken bij het herkennen van bruikbaarheidsproblemen in gebruiksonderzoeken en het bedenken van een manier om inconsistenties aan te pakken. Daartoe moet

- 1 een procedure ontwikkeld worden om analyses goed te documenteren en inspecteerbaar te maken,
- 2 onderzocht worden in welke mate deze procedure mogelijke bronnen van inconsistentie blootlegt en in staat is om inconsistenties voortkomend uit vermoeidheid, verminderde oplettendheid en gebrek aan concentratie terug te dringen,
- 3 de procedure toegepast worden in een vergelijkende studie waarin het effect van het gebruik van prototypes (in plaats van werkende producten) in gebruiksonderzoeken gedetailleerd onderzocht wordt.

1 Ontwikkeling van de (Slim)DEVAN procedure voor data analyse

In hoofdstuk 2 wordt de DEVAN (DEtailed Video ANalysis) procedure geïntroduceerd. DEVAN zorgt ervoor dat de data-analyse van gebruiksonderzoeken goed gedocumenteerd kan worden en inspecteerbaar is. De inspecteerbaarheid van analyses wordt in DEVAN gerealiseerd door interacties in tabellen op verschillende abstractieniveaus weer te geven. Resultaten van de verschillende stappen van de analyse verwijzen direct naar gegevens in deze tabellen. Om de consistentie in de overzichtstabellen te bevorderen, worden de interacties in segmenten opgedeeld met behulp van drempeltijden voor de pauzes tussen handelingen. De analist wordt geholpen om bij het vaststellen van gebruiksproblemen consistent te blijven, door middel van een controlelijst met waarneembare signalen voor problematische gebeurtenissen ('signaalgebeurtenissen'). Moeilijkheden die in de interactie kunnen optreden, worden op een gestandaardiseerde manier beschreven. Deze beschrijvingen verwijzen naar specifieke gegevens in de overzichtstabellen met interacties.

In hoofdstuk 3 wordt de ontwikkeling beschreven van SlimDEVAN, een variant van DEVAN. SlimDEVAN is bedoeld als een eenvoudiger en minder tijd vergende versie van DEVAN. Om dat te bereiken, zijn de verschillende abstractielagen uit de overzichtstabellen met interacties verwijderd en wordt niet voorgeschreven hoe de interacties in segmenten moeten worden opgedeeld. De controlelijst met signaalgebeurtenissen

wordt net zo ingezet als bij DEVAN en behalve interactieproblemen kunnen nu ook mogelijke oorzaken van die problemen gerapporteerd worden. De vorm van rapporteren is eenvoudiger geworden dan in DEVAN, zonder dat daarbij de verwijzingen naar gegevens in de overzichtstabellen met interacties verloren zijn gegaan.

2 Onderzoek naar inspecteerbaarheid en consistentie

In de hoofdstukken 2 en 3 worden onderzoeken gerapporteerd waarin met behulp van (Slim)DEVAN verscheidene vraagstukken op het gebied van inspecteerbaarheid en consistentie onder de loep zijn genomen. In hoofdstuk 2 wordt een onderzoek gerapporteerd waarin de inspecteerbaarheid van DEVAN werd beoordeeld door vergelijking van verschillende lijsten met interactieproblemen. Deze lijsten werden door twee onafhankelijk van elkaar werkende analisten (de bedenkers van DEVAN) afgeleid van video-opnamen van steeds dezelfde onderzoekssessies. In hoofdstuk 3 wordt onderzoek beschreven naar inspecteerbaarheid en consistentie waarin interactieproblemen zoals gerapporteerd door drie professionele 'usability' labs, vergeleken werden. De labs onderzochten alle drie hetzelfde product met behulp van een gezamenlijk ontwikkeld testprotocol. Elk lab analyseerde de eigen gegevens twee keer: eerst volgens de eigen standaardprocedure en, na een paar weken, volgens de procedure van SlimDEVAN. In een vervolgstudie (ook beschreven in hoofdstuk 3) werd een deel van de gegevens van één van de labs opnieuw twee keer geanalyseerd, maar nu door andere analisten, die in beide gevallen SlimDEVAN gebruikten.

Binnen-analist consistentie en inspecteerbaarheid De eerste en de tweede analyse van elke analist werden met elkaar vergeleken om elementen in de manier van rapporteren van problemen te onderkennen die van belang waren om inspecteerbaarheid te realiseren, om de consistentie tussen de twee analyses van elke analist te bepalen en om de oorzaken van inconsistenties te achterhalen. Er kon een lijst opgesteld worden met eigenschappen van probleemrapportages die bijdragen aan de inspecteerbaarheid van die rapportages. De binnen-analist consistentie bleek aanzienlijk te variëren. Een analyse van de oorzaken van binnen-analist inconsistenties wees uit dat de binnen-analist consistentie hoger had kunnen zijn, als er geavanceerdere (geautomatiseerde) hulpmiddelen waren gebruikt voor het loggen en in tekst vastleggen van het gedrag van gebruikers in combinatie met nauwkeuriger criteria voor het definiëren van problemen. Een vergelijking van de scores voor binnen-analist consistentie laat een tendens zien van hogere binnen-analist consistenties voor die gevallen waarin SlimDEVAN gebruikt werd in beide analyses. Verder onderzoek is nodig om na te gaan in hoeverre dit toe te schrijven is aan het gebruik van SlimDEVAN.

Tussen-analisten consistentie en inspecteerbaarheid Lijsten met bruikbaarheidsproblemen en lijsten met moeilijkheden van de verschillende analisten werden vergeleken om het beoordelaarseffect te bepalen, om vast te stellen of de consistentie tussen analisten verhoogd zou kunnen worden door (Slim)DEVAN te gebruiken en om te achterhalen waaruit inconsistenties tussen analisten voortkomen. Per vergelijking varieerde

Wat is 't probleem?

Onderzoek naar het herkennen van bruikbaarheidsproblemen in gebruiksonderzoeken

de grootte van het effect aanzienlijk, maar in alle vergelijkingen was een effect aanwezig. De conclusie moest dan ook zijn dat het beoordelaarseffect niet makkelijk geëlimineerd kan worden. De resultaten wijzen erop dat de verschillen in beoordelaarseffect veroorzaakt zijn door het kiezen van verschillende eenheden om het effect te berekenen en door verschillen in de geoefendheid van de analisten met (Slim)DEVAN. Het bleef onduidelijk of gestructureerde, inspecteerbare probleemrapportages kunnen leiden tot een verhoogde tussen-analisten consistentie. Het onderzoek uit hoofdstuk 2 liet zien dat onduidelijkheden in de criteria voor het definiëren van problemen bijdragen aan het beoordelaarseffect. Vastgesteld is dat tussen-analisten inconsistenties verder veroorzaakt werden door verschillen in de opvattingen, waarden en voorkeuren van de analisten, door de hulpmiddelen voor het observeren van het gedrag van gebruikers en door het, als gevolg van vermoeidheid, verminderde oplettendheid en gebrek aan concentratie, niet opmerken van ontbrekende handelingen. Het zou interessant zijn om nader te onderzoeken of en in welke mate de consistentie hoger zou worden in die gevallen waarin het duidelijk is welke waarden van belang zijn voor het bedrijf dat opdracht geeft voor het onderzoek, en waarin met deze waarden rekening wordt gehouden bij het bepalen van het beoordelaarseffect. Als aanpak voor het omgaan met de verschillen in opvattingen, waarden en voorkeuren van analisten wordt voorgesteld de mogelijkheid te overwegen om teams van analisten de gegevens gezamenlijk te laten analyseren. Wanneer analisten in teams werken, zou dat als voordeel kunnen hebben dat de verschillende gezichtspunten al tijdens de analyse expliciet gemaakt worden en dan ook besproken en mogelijk opgelost kunnen worden, nog voordat de definitieve lijsten met problemen worden opgesteld.

3 DEVAN toegepast op een vergelijkend onderzoek met prototypes

In hoofdstuk 4 wordt beschreven hoe DEVAN is toegepast in een vergelijkend onderzoek. Het doel van dit onderzoek was te ontdekken welke invloed de eigenschappen van prototypes en aanpassingen in de testsituatie die gepaard gaan met het gebruik van prototypes, hebben op het gedrag van een proefpersoon en op de lijst met problemen die het onderzoek oplevert. Er werd onderzoek gedaan met een werkend product en met twee verschillende soorten prototypes daarvan. De drie soorten resultaten werden onderling vergeleken. In het kader van dit proefschrift had het onderzoek twee doelen, namelijk vaststellen of analyse volgens DEVAN zodanig inspecteerbaar is dat deze procedure bruikbaar is in het vergelijkende onderzoek, en nagaan of de resultaten van een analyse volgens DEVAN succesvol te combineren zijn met een foutentaxonomie bij het zoeken naar oorzakelijke verbanden.

Het vergelijkende onderzoek bleek aan beide doelen te beantwoorden. Gedetailleerde analyse van de problemen bracht aan het licht hoe het gedrag van gebruikers beïnvloed werd door de eigenschappen van de prototypes en door de rol van de proefleider. Ook werd duidelijk dat bij summatieve prestatie-maten het gevaar bestaat dat interessante verschillen in het gedrag van gebruikers en in door hen ondervonden moeilijkheden uitgemiddeld worden. De gevonden verschillen bleken voornamelijk veroorzaakt

te zijn door elementen waarin dit product zich onderscheidde van de producten van concurrenten.

4 Conclusie

In dit proefschrift is aangetoond dat inconsistenties bij het analyseren van de gegevens van gebruiksonderzoeken heel hardnekkig zijn. Als een manier om de meer hardnekkige inconsistenties aan te pakken, wordt voorgesteld de analyses inspecteerbaar te maken. Om de minder hardnekkige inconsistenties terug te dringen, worden geavanceerde (geautomatiseerde) hulpmiddelen voor het observeren van gedrag voorgesteld en verder ook nauwkeuriger criteria voor het definiëren van problemen. Aangeraden wordt om nader onderzoek te doen naar de gevolgen die inconsistenties hebben in de productontwikkelingspraktijk. Uiteindelijk zal dit ertoe leiden dat beter begrepen wordt hoe een goed gebruiksonderzoek moet worden opgezet. Het gevolg daarvan zal weer zijn dat minder gebruikers mopperen: *“Wat is ’t probleem? Waarom doet dit ding niet wat ik wil?”*

Arnold Vermeeren, maart 2009

Wat is 't probleem?

Onderzoek naar het herkennen van bruikbaarheidsproblemen in gebruiksonderzoeken

Dankwoord / Acknowledgements

In de loop van dit project hebben veel mensen op een directe of indirecte manier bijgedragen aan de totstandkoming van dit proefschrift. Het is me steeds duidelijker geworden dat onderzoek niet iets is dat je alleen doet. Hieronder wil ik een aantal mensen bedanken.

Huib de Ridder, jouw enthousiasme als promotor en 'chef' was geweldig. Het was precies wat ik nodig had om het project uiteindelijk tot slagen te brengen. Je hebt me altijd het gevoel gegeven dat je achter me stond en dat je er vertrouwen in had dat ik het tot een goed eind kon brengen. Van jou heb ik geleerd wat onderzoek doen is en wat kwaliteit betekent. Ik wil je daar heel hartelijk voor bedanken.

Ans van Doorn, je hebt me vaak verbaasd met de snelheid waarmee je mijn manuscripten van commentaar hebt voorzien, maar ook met de nauwgezetheid en gedetailleerdheid waarmee je dat deed. Altijd bleef je kritisch, maar ook bemoedigend en opbouwend. Dank je wel daarvoor.

Jans Aasman, jij hebt mijn promotiewerk in gang gezet toen er na het overlijden van Rudy den Buurman en het vertrek van Kine Sittig niemand meer bij mijn onderzoek betrokken was. Jouw enthousiasme was daarbij van doorslaggevend belang. Je eerlijkheid en directheid waardeer ik zeer. Ik ben je dankbaar voor de bijdrage die je hebt geleverd, die was erg belangrijk voor me.

Rudy den Buurman (†) en *Kine Sittig* wil ik bedanken voor het mogelijk maken van en het zelf richting mogen geven aan het onderzoek dat de aanleiding was voor dit project.

Tilde Bekker, ik heb je niet voor niets als paranimf gevraagd: Je bent een vriendin en een fijne collega. Al heel lang werken we op onderzoeksgebied samen en zien we elkaar ook buiten het werk regelmatig. Jij zette me op het spoor van het artikel van Lavery and Cockton (1999) dat van doorslaggevende betekenis werd voor dit proefschrift en maakte me attent op het COST294 project waaruit belangrijke contacten zijn voortgekomen, niet in het minst voor de samenstelling van de commissie. Daar wil ik je voor

What's the problem?

Studies on identifying usability problems in user tests

bedanken. Maar ik wil je ook bedanken voor die onvoorspelbare, onbedaarlijke lachbuien die zo lekker relativerend werken. Ik hoop nog vaak met je te kunnen samenwerken.

Karin den Bouwmeester, Jettie Hoonhout, Wolmet Barendregt en Ilse van Kesteren wil ik bedanken voor hun bijdrage aan de ontwikkeling van en het nadenken over DEVAN.

Karin, wij hebben mede in het kader van jouw afstudeerproject samen DEVAN ontwikkeld. Dat was een cruciale stap in dit project. Ik vond het een plezierige samenwerking. Jettie, dankzij jou is Karin bij ons terecht gekomen. Wolmet en Ilse, jullie gebruikten DEVAN en pasten die aan. Dat leverde nuttige nieuwe gezichtspunten en aanpassingen op.

The members of the reading committee I would like to thank for their comments on the draft of this thesis. Your comments have certainly improved the thesis. Especially, I would like to thank *Gilbert Cockton* for his extensive, useful comments and suggestions, *Kaisa Väänänen-Vainio-Mattila* for her suggestions on revising the introductory chapter, and *Kasper Hornbæk* for his insightful thoughts that sharpened chapter 5.

Çigdem Erbuğ, Martin Maguire, Evren Akar and Ali Berkman I would like to thank for conducting the user tests that formed the basis for chapter 3, and especially for being willing to use SlimDEVAN and go through all your video tapes again. It was crucial to this thesis. Thank you very much; I have good memories to the enjoyable times we had in Ankara. Thanks also to *Deana McDonagh, Pelin Gültekin and Zeynep Karapars* for their contribution to the user tests, and *Nigel Bevan* for his contributions in the early phases of the collaborative project.

Elyon Dekoven and Jelle Attema wil ik bedanken voor hun essentiële bijdrage aan de totstandkoming van het onderzoek van hoofdstuk 3. Jelle, bedankt dat je me gevraagd hebt deel te nemen aan het project waaruit hoofdstuk 3 is voortgekomen. Elyon, thank you very much for bringing up the idea to use SlimDEVAN in the international project so that I could use it for my thesis. Clearly, this was a great idea.

I would like to thank *Philips Netherlands B.V.*, particularly *Edwin van Vianen*, for providing me with a prototype of the TV-video combination, and *Paremion* for building the software prototype. *Jacqueline van Heist* wil ik bedanken voor het uitvoeren van de onderzoeken met de TV-video combinatie. Alle deelnemers aan de tests en alle studenten en student-assistenten die op een of andere manier aan de onderzoeken hebben bijgedragen wil ik ook hartelijk bedanken.

Sjaak van Asten en Arend Harteveld (†) wil ik bedanken voor de inrichting van de laboratoria en de ondersteuning bij de tests. Jullie expertise was onmisbaar en het was een plezier om met jullie te werken.

Theo Boersema, ook jou heb ik gevraagd paranimf te willen zijn. We hebben altijd plezierig contact tijdens de pauzes en het werk. Je bent een fijne collega, op wie ik altijd kan bouwen. Ik wil je in het bijzonder bedanken voor je hulp en het meedenken in de laatste fase van dit project.

Theo Rooden, de omslag en vormgeving van dit proefschrift en enkele figuren daarin zijn door jou ontworpen. Ik wist dat ik er op kon rekenen dat het een mooi geheel zou worden. Het was een plezier om met je te werken. Je bent voortvarend in je aanpak,

betrouwbaar, behulpzaam en levert mooi werk af. Maar dat dat zo is wist ik eigenlijk al uit de tijd dat wij nog collega's waren.

Gedurende de loop van dit project heb ik op de faculteit met veel mensen contact gehad. Ik heb 'tussen de bedrijven door' fijne gesprekken gehad met *Elif, Annelise, Stella, Marijke, Armağan, Niels* en met andere collega's; de mensen van het secretariaat gaven steeds plezierige en efficiënte ondersteuning. Verder wil ik ook de mensen bedanken met wie ik meer inhoudelijk over (mijn) onderzoek van gedachte heb mogen wisselen. Op het gevaar af dat ik een aantal van hen vergeet noem ik toch: *Heimrich, Jasper, Adinda, Marlies, Jozina, Paula* (†), *Guus, Gert-Jan, Nicole, Faye en Mieke*.

Met *Cees, Henk, Theo, Frits* en *Arend* (†) bracht ik vele koffie- en lunchpauzes door. Dat was erg belangrijk om af en toe op adem te komen of mijn ei kwijt te kunnen als er dingen mis gingen. Maar bovenal waren het gezellige pauzes.

Naast mensen op het werk zijn er nog andere mensen die mij inspiratie geven op allerlei gebied. Mijn taiji-lerares *Zhang Zhijie* en *Wang Xiali* wil ik bedanken voor de dingen die ik van hen leer, waar ik mijn hele leven wat aan zal hebben en die me helpen in balans te blijven in moeilijke tijden. I would like to thank my haiku teachers in Japan, *Sugawa sensei* and *Dhugal J. Lindsay*, who taught me to look at the world with different eyes and to discover 'truths of human existence in nature'. Ook aan al die anderen die ik tegenkwam op de haiku- en taijiweg ben ik dank verschuldigd. Op de faculteit geldt dat vooral voor de groep van mensen die me de gelegenheid gaven en geven om in de lunch-taiji bijeenkomsten mijn enthousiasme over te dragen. Dat gaf me tevens de mogelijkheid me op die momenten van mijn stress te ontdoen: *Agnes, Tjamme, Hélène, Pieta, Marieke, Marco, Els, Marlies, Elyon, Xi, Armağan, Petra, Maria*, dank jullie wel daarvoor.

Rutger Kopland wil ik bedanken voor het feit dat ik zijn gedicht zonder voorwaarden anders dan bronvermelding mocht gebruiken; *Willem Groenewegen* dank ik voor het belangeloos werken aan de Engelse vertaling van het gedicht en het ter beschikking stellen ervan.

Tenslotte wil ik de mensen bedanken die me het meest nabij zijn, mijn familie. Vaak las ik in een proefschrift woorden als 'zonder jullie steun was dit nooit gelukt'. Ik weet nu hoe waar en gewenst dit soort uitdrukkingen kunnen zijn (zie ook stelling 10). *Pa en ma, Huub en Sipkje, Ome Ad*, ook jullie hebben ieder op jullie eigen manier aan deze promotie bijgedragen. Ik waardeer dat zeer. *Marion, Jeroen en Maaïke*, dat jullie geweldig zijn wist ik al, maar in de laatste weken van dit project is dat nog eens extra bevestigd. Die weken waren ook voor jullie een beproeving, maar jullie hebben me steeds geweldig bijgestaan. Ik wil jullie duizendmaal danken, ik hou van jullie.

In al die jaren zijn er ongetwijfeld nog veel meer mensen geweest die op een of andere manier bijgedragen hebben aan dit proefschrift. Ook hen wil ik hartelijk bedanken.

Arnold Vermeeren
Maart 2009

What's the problem?

Studies on identifying usability problems in user tests

What's the problem?

Studies on identifying usability problems in user tests

Curriculum Vitae

Arnold Vermeeren was born on 27 November 1959 in Oss. In 1987 he received his Master's degree in Industrial Design Engineering at Delft University of Technology. Since then he works as an Assistant Professor at that faculty, initially to set up the first user interface laboratory of the faculty. In the years that followed he was involved in teaching and in conducting usability research in various international projects collaborating with academic institutes and industry. For two years he was manager of a project in which the faculty was asked to conduct various commercial usability tests for a large bank in the Netherlands. Currently he is involved in projects on evaluating user experiences of software and interactive products and in teaching courses on usability, user experience and user-centered design.

“Ik streef er naar niet raar te vinden wat mensen doen, het niet te betreuren of te veroordelen, maar het te begrijpen”.

Benedictus de Spinoza

(Tractatus Politicus, Hoofdstuk 1, deel 4)

What's the problem?

Studies on identifying usability problems in user tests

List of publications related to the thesis

Vermeeren A.P.O.S., den Bouwmeester K., Aasman J., and Ridder H. de (2002). DEVAN: A tool for detailed video analysis of user test data. *Behaviour & Information Technology*, 21 (6), pp 403-423 (chapter 2 of this thesis).

Vermeeren A.P.O.S., Attema J., Akar E., Ridder H. de, Doorn A.J. van, Erbug C., Berkman A.E. and Maguire M.C. (2008) Usability Problem Reports for Comparative Studies: Consistency and Inspectability. *Human-Computer Interaction* 23 (4), pp 329-380 (chapter 3 of this thesis).

Vermeeren A.P.O.S., Koenderink-van Doorn A.J. and Ridder H. de (2006). Reliability of a checklist-based user test data analysis procedure. In RN Pikaar, EAP Koningsveld & PJM Settels (Eds.), *Proceedings IEA2006 Congress*. Amsterdam: Elsevier (follow-up study in chapter 3 of this thesis).

Vermeeren A.P.O.S., de Ridder H., van Doorn A.J. (submitted). Identifying usability problems in product tests: effects of using cardboard and software prototypes. (submitted to *Journal of Usability Studies*, October 2008) (chapter 4 of this thesis).

Vermeeren, A.P.O.S., Kort J., Cremers, A.H.M. and Fokker J., (2008) Comparing UX Measurements, a case study. *5th COST294-MAUSE Open Workshop on Valid Useful User Experience Measurement (VUUM)*, Reykjavik, Iceland, pp 72-78.

Vermeeren A.P.O.S., Koenderink-van Doorn A.J. and Ridder H. de (2006). What, how and why of making user test data analyses explicit. In RN Pikaar, EAP Koningsveld & PJM Settels (Eds.), *Proceedings IEA2006 Congress*. Amsterdam: Elsevier.

Erbug Ç., Attema J., Vermeeren A.P.O.S., Maguire M., Berkman A.E., Akar E. and McDonagh D. (2005). Usability testing: a collaborative approach. In *HCI international 2005: 11th international conference on human computer interaction*, 22-27 July 2005, Las Vegas, Nevada, USA. Lawrence Erlbaum Associates.

What's the problem?

Studies on identifying usability problems in user tests

Vermeeren A.P.O.S. (2004). Experiences with conducting contract research for a large banking company, a case study. In C Erbug (Ed.), *Usability testing: methods, experiences, achievements*, Ankara: METU Faculty of Architecture Press, pp. 161-172.

Vermeeren A.P.O.S. and Soleymani F. (2004). Integrating user research activities in the design process, a case study: the Virtual Integration Counter. In C Erbug (Ed.), *Usability testing: methods, experiences, achievements*. Ankara: METU Faculty of Architecture Press, pp. 99-108.

Vermeeren A.P.O.S. (2004). Structured video analysis of user test data, using the DEVAN tool. In C Erbug (Ed.), *Usability testing: methods, experiences, achievements*, Ankara: METU Faculty of Architecture Press, pp. 123-140.

Vermeeren A.P.O.S. (2004). The evaluator effect in user testing. In C Erbug (Ed.), *Usability testing: methods, experiences, achievements*, Ankara: METU Faculty of Architecture Press, pp. 189-199.

Kesteren I.E.H. van, Bekker M.M., Vermeeren A.P.O.S. and Lloyd P.A. (2003). Assessing usability evaluation methods on their effectiveness to elicit verbal comments from children subjects. In S MacFarlane, T Nicol, JC Read & LC Snape (Eds.), *Proceedings of the 2003 conference on Interaction design and children: Small users - big ideas*, New York: Association for Computing Machinery, pp. 41-49.

Vermeeren A.P.O.S., Kesteren I.E.H. van and Bekker M.M. (2003). Managing the evaluator effect in user testing. In M Rauterberg, M Menozzi & J Wesson (Eds.), *Human-computer interaction INTERACT '03*. Amsterdam: IOS Press, pp. 647-654.

Vermeeren A.P.O.S. (1999) Designing Scenarios and Tasks for User Trials of Home Electronic Devices. In W.S. Green, P.W. Jordan (eds.) Chapter five: *Human Factors in Product Design: Current Practice and Future Trends*, London: Taylor & Francis, pp 47-55.

