Towards Systematic Usability Verification

Max Möllers
RWTH Aachen University
52056 Aachen, Germany
max@cs.rwth-aachen.de

Jonathan Diehl
RWTH Aachen University
52056 Aachen, Germany
diehl@cs.rwth-aachen.de

Markus Jordans
P3-Solutions GmbH
Dennewartstr. 25-27
52068 Aachen, Germany
jds@p3-solutions.de

Jan Borchers
RWTH Aachen University
52056 Aachen, Germany
borchers@cs.rwth-aachen.de

Abstract

Although usability is the core aspect of the whole HCI research field, it still waits for its economic breakthrough. There are some corporations that are famous for their usable products, but small and medium-sized businesses tend to prefer features over usability. We think, the primary reason is that there are no inexpensive methods to at least prevent huge design flaws. We propose the use of test specifications. Once defined for a domain, these allow non-usability experts to systematically verify the usability of a given system without any users involved. We picked a sample domain with some basic tasks and found strong indication of our hypothesis: test specifications can be applied by non-experts and are able to find major design flaws. Future work will extend this method to more complex tasks and evaluate the economic benefit.

Keywords

Usability verification, quality management, analysis methods, quantitative usability evaluation, performance metrics, usability testing and evaluation.

ACM Classification Keywords

H5.2 User Interfaces: Evaluation/methodology.

Introduction

Designing interfaces and new products that are of interest and use to a client are typical challenges for a company. To ensure that these products fulfill certain standards, quality management processes are established. Such processes require tools for quality assessment. For functional assessment this is done by, e.g., using functional test specifications, where test specifications contain a step-by-step instruction lists that are checked by an employee after the design process to ensure that industry standards have been met.

Our vision is to use test specifications to ensure high usability standards in products. Having a test specification, companies can easily test for the major design flaws. This method will presumably return inferior results in comparison to a full-fledged user centered design process, but non-usability experts can apply the test specification without the need for elaborate user-tests and huge numbers of users. We assume that the lower cost of this testing method will drastically increase the willingness of companies to include usability considerations into their product design or in buying decisions of third-party products – think of network providers deciding which latest mobile phone to buy. Another application field could be product line development. Here, small design chances can break the usability, yet companies might tend to do a user test only for every new major release. With test specifications *checking* for problems, the usability engineers can do their job: fixing problems or, even better, designing for problem free interfaces.

Our suggested method is to gather the expert knowledge of a particular product domain and boil it down to a test specification. Once a company has acquired such a test specification for its domain, it can apply it to its own or third party products to identify their design flaws.

In this paper, we will outline previous research in this field, describe our methodology, describe a showcase domain, and show the progress of our evaluation.

Related Work

There are many long-established methods for usability verification. None of these methods, however, meets our requirements of not employing test users or usability experts for their performance.

User tests are a classic technique for usability evaluation [2] and considered by many as state of the art. User tests always require test users and usability experts, which makes them impractical for our purposes. Suggestions for reducing the cost, such as by [4], help widening the applicability of this method but cannot remove the cost of employing users or experts entirely.

Cognitive Walkthrough [2] is another technique, which is used by usability experts to evaluate a system without users. Here, the evaluators walks through the system and analyze the visibility, accessibility, comprehensibility, and given feedback for each step. The method depends on usability expertise among the evaluators and will yield questionable results if performed by non-experts.

Model-based methods, like GOMS [3] or Petri Nets [5], model user behavior on interactive systems and allow quantitative usability evaluation without employing test users. These methods, however, mostly consider task performance times or reach ability and cannot test visual (labels, icons, etc.), ergonomic, or comprehension aspects of the usability of a system.

Methodology

We divide the process into four steps:

1. Identify the target domain.
2. Gather usability knowledge.
3. Translate knowledge into the test specification.
4. Perform the test.

The first three steps will result in a test specification for the domain, which can then be used to run the tests on a number of similar products. Consequently, it is acceptable if the first three steps require high effort of usability experts and test users, because they must only be performed once and are independent of the number of products to be tested. This could also be done by one consortium and then shared in the domain – as it often happens for other industry standards. We are going to give some examples for this process from our sample domain in *smaller italics*.

1. Domain Identification

The first step is to identify the domain of the product, which can be done through a classic task analysis [2]. *The sample domain is mobile phones with people from 25-35, all genders, and experience in using mobile phones. The task is to send a short message to a contact. One strategy is to go to the message menu, write the message, select the contact from address book, and send the message.*

We first identify users of the domain through market research or interviews. Then for each user type, we identify relevant tasks and decompose them into fine-grained subtask resulting in strategies, which we write down as state machines (cf. Figure 1). For the necessary data collection, we have two resources:

- A user test with a broad spectrum of products in the domain, where the users should try to achieve the identified tasks. This shows how people *do* use current products.
- An analysis of existing user interfaces in the domain, e.g., by employing the cognitive walkthrough method [4]. This shows how people *can* use current products.

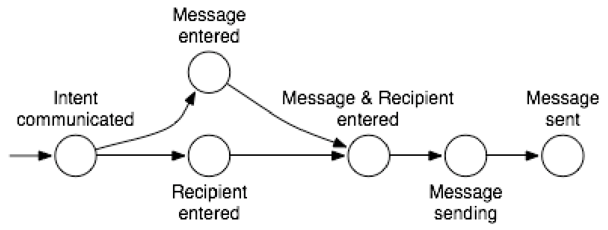

Figure 1. State machine for the task of sending a short message on a typical mobile phone

1.1 Domain-Specific Design Problems

The user tests for the identification of strategies have another benefit if done right. Because the test users will not be familiar with every product, unexpected behavior of the user interface will lead to confusion or worse. Employing the think-aloud method [2], the test team can track these design flaws and compile a list of design problems. The most common entries in this list

Precondition	Action Step	Action Result	A	B	C
Idle screen active	Enter menu, measure time between click and complete screen update	Menu active	Instantaneous	Slight delay (time <= 200ms)	Noticeable delay (time <= 500ms)
Address book active (from Idle screen)	Initiate Write SMS function from first contact	Write SMS active, recipient selected	Interactions <= 3 Label: Write/Create/New Message	Interactions <= 6 Label: Write/Create/New Message	Interactions <= 10

Table 1. Two tests from our example: The first ensures that the menu and the second that the 'Write SMS' function can be accessed efficiently.

are what we call *key design problems*. Performing well at these key design problems should be crucial for a usable product in this domain. To easily identify good performing products, one should use measurable properties. For not measurable properties, e.g., wording of menu entries, one should compile a list of domain-specific common terms to compare to. *For our sample domain, mobile phones, we found in our study that the consistent wording of menu entries, the length of interaction, and the awareness of the system's mode are of great importance.*

2. Domain-Specific Guidelines

Although we already identified the key design problems, we must not overlook the parts of the product's design that work. Only checking for the key design problems, the test specification could give good grades for a product with none of these problems, but other even bigger problems. Fortunately, good designs are often captured in guidelines (e.g., [6]), and design patterns (e.g., [7]). Knowing the domain, we can gather existing guidelines close to and relevant for our domain. Typically, there will not be *the* book about our domain, so we need to use other resources.

Sometimes there are vendor-specific interface guidelines (e.g., [1]), which we will need to abstract, but we can always refer to some meta guidelines and interpret them for our domain.

3. Translation into Test Specifications

Having practical insight on the one hand and expert knowledge on the other, the real challenge now is to create the test specifications. A test specification consists of several tests for which we suggest the following format:

- *Precondition* describes the state the system has to be in when the test starts.
- *Action Step* describes what the tester has to do.
- *Action Result* describes what the result should be.
- *Grading (A, B, C,...)* lists several possible properties of different implementations. The better the implementation regarding key design problems and usability aspects, the higher the grade. We used a scale from A to C for passed tests with A the best and F for a failed test.

Constructing the tests works as follows. We review each strategy and its state machine. For each transition, we check our practical and theoretical knowledge. We create a test whenever one of the key design problems could occur in the transition, or when a guideline is applicable. The different grading entries are constructed from possible interfaces from the domain. These possibilities are now rated according to the previous acquired guidelines with a strong focus on the key design problems identified earlier. Qualitative properties are rated according to their commonness – users have domain-specific knowledge and will prefer known terms and conceptual models. *Table 1 shows two typical tests. The first test is based on the guideline "Make the user* *interface responsive", which is found in most high-level guidelines and is applicable to our domain as well. The second test originates from our finding that the key design problems include uncommon wording menu entries and unnecessary long interaction sequences. The labels were taken from a market study.*

4. Applying the Test Specification

After creating the test specification, we can now apply it to a new product to see how it passes our tests. To do so, a tester takes the product in question and the test specification and executes each test. This tester does not need to have a deep understanding of usability, or several users to survey in front of him. Failed tests point out major problems in the user interface. How to summarize all different grades into one usability "score" is another topic of research, but an overall well-rated device is usable with regard to the test specification.

Preliminary Study

As one step in supporting our claims, we had to make sure that this idea works in at least simple cases. We first needed a test specification.

Domain and Test Specification

We chose the domain of mobile phones with users of all ages and genders, who use the mobile phone as consumers. The proposed sample task was to send a short message to a contact in the address book. We ran a user test with a small group of five people with three mobile phones from different vendors, resulting in key design problems and common user strategies. In addition, we evaluated the current user interface of ten different mobile phones to also collect vendor-proposed strategies. We then created state machines and tests based on these. In parallel, we studied the literature to identify suitable guidelines, resulting in 217 candidate guidelines and patterns. After creating tests from our state machines and guidelines, we ended up with 94 unique tests – our test specification for this domain.

Evaluation

To prove the validity of our approach, we would have to show at least two things:

- (H1) The test specification gives results similar to the state-of-the-art usability verification method – a complete user test.
- (H2) The test specification can be used by non-usability experts, and its results are not influenced significantly by the tester's individual opinion or experience.

Within our preliminary study, we evaluated these hypotheses with strong indication that they are true for our sample domain.

For H1, we independently ran a user test and a verification process based on our test specification. We chose five current mobile phones and observed ten different users using each of them. Each participant was given three tasks:

- Send a specific short message with some special characters from the *send message* function to a contact in the address book.
- Send a longer message, including a formatted time string, directly from a contact in the address book.
- Send a number as a message to another national phone number.

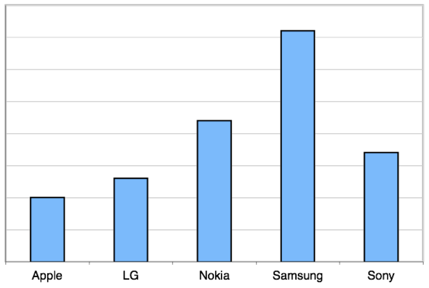

Figure 2. Results of the user test showing the number of problem occurrences for each device.

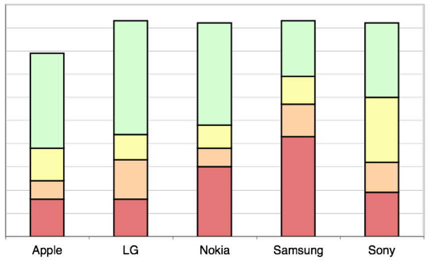

Figure 3. Results of the specification test showing the number of test cases graded in each category (from top to bottom: A, B, C, F) for all devices.

Meanwhile, the test specification was given to five different participants. The overall group was of similar age and gender composition, mostly from a technical background. Each participant tested one of the devices.

As we can see in Figures 2 and 3, the devices that led to problems in the user test, also failed similarly often in the test specification, indicating the validity of H1.

For H2, three more testers tested one of the devices using the test specification. All testers agreed on whether to pass or fail a device in 84% of the tests. Considering the early state of our test specification, we are confident that more precise test formulations will improve this ratio. Conclusively, these four non-usability experts came to very similar results, which indicates the validity of H2.

Outlook

Our preliminary study gives enough evidence to pursue this topic in more detail. One open question is whether the method also works for more complex tasks in more general interaction spaces than the menu.

Further, we need to investigate the cost of creating and maintaining a full test specification in a realistic situation to be able to judge its applicability for quality management.

To achieve this and truly show the validity of our approach, we need to run large evaluations of the method in a realistic environment, testing real products in the market and comparing the results to more user tests, run by independent organizations.

Acknowledgements

Special thanks go to Volker Schmidt for his dedicated work on the user tests. We thank all of our test participants. This work was funded in part by the German B-IT Foundation.

Example citations

[1] Apple Human Interface Guidelines. Apple Inc., 2008.

[2] Dix, A., Finlay, J., Abowd, G., and Beale, R. Human-computer interaction. Prentice-Hall, Inc., 2004.

[3] John, B. E. and Kieras, D. E. The GOMS family of user interface analysis techniques: comparison and contrast. ACM Trans. Comput.-Hum. Interact., 1996, 3(4),320--351.

[4] Nielsen, J. Usability Engineering. B&T, 1994.

[5] Petri, C. A. Concepts of net theory. Mathematical Foundations of Computer Science: Proceedings of the Symposium and Summer School, 1973, 137--146.

[6] Shneiderman, B. Designing the User Interface (4th Edition). Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA2005.

[7] Tidwell, J. Designing Interfaces. O'Reilly Media, Inc., 2006.