Tähän monisteeseen on valittu otos lisensiaattityöstä
Riihiaho, S. (2000) Experiences with usability evaluation methods. Licentiate's thesis. Helsinki University of Technology, Department of computer science and engineering. 113 p.

1 Experience with usability inspection

The most common usability inspection method in our usability evaluations at Helsinki University of Technology is the heuristic evaluation. We have also applied cognitive walkthrough, but not as often as heuristic evaluation. We have not applied any automatic inspection methods, and the formal methods, such as GOMS, have been left to little attention in our work.

Usually we do the inspections in two phases. In the first phase, we try to get familiar with the system, search for the most severe problems, and think of ways to test its usability. After the tests, we have more knowledge of the system and its use, so we are able to deepen our inspections.

The next sections present some of our experience with heuristic evaluation, cognitive walkthrough and GOMS. The examples try to give an overview of what kind of problems these methods reveal and what sort of input they could give to the development process.

1.1 Heuristic evaluation

Our usability group has done a heuristic evaluation to most of the systems that we have evaluated. The systems include professional software systems, web applications, consumer electronics and professional equipment. Usually, we do a heuristic evaluation to a system while we are familiarising ourselves with the system and planning a usability test. After the tests, we sometimes make a new evaluation to check if we missed something in the first round. At this point, the results of the usability tests may easily affect the results of the heuristic evaluation.

Heuristic evaluation is easy to teach and easy to learn in general. Therefore, it is a very popular method in students' course assignments. The students are almost novice evaluators as they conduct the evaluations, so they conform to the heuristics quite heavily and leave other issues, such as support to users' tasks, to little attention. As the evaluators' skills improve, the list of heuristics changes and the whole method gradually changes into a combination of heuristic evaluation and cognitive walkthrough.

The output of our heuristic evaluations is a list of usability problems. We present the general problems affecting the whole system first, and after that, problems relating to certain dialogues or certain functions of the system. The problems are in order of priority, most severe first. Usually, we use only three levels of severity to highlight the difference between the levels: major, disturbing and minor problems. If the system is on the stage of early prototyping and it has many major and disturbing problems, some cosmetic problems might be left out of the problems, so that the list of the problems is not overwhelming.

1.1.1 Ways to present the results

The process of heuristic evaluation is straightforward. Presenting the results in a clear and readable form is not as easy. In the course assignments we have given, students easily present the results in a textual list with poor specifications if not otherwise instructed. A good way to present the results is to show the dialogue, attach the problems to the dialogue and specify the problems in a textual list in a table. The problems should be in the order of priority: most severe first and the minor problems last. The heuristics that the problem violates should also be shown in the table. Figure 1 shows an example how to link the problems to the dialogue. Table 1 presents the problems in more detail.

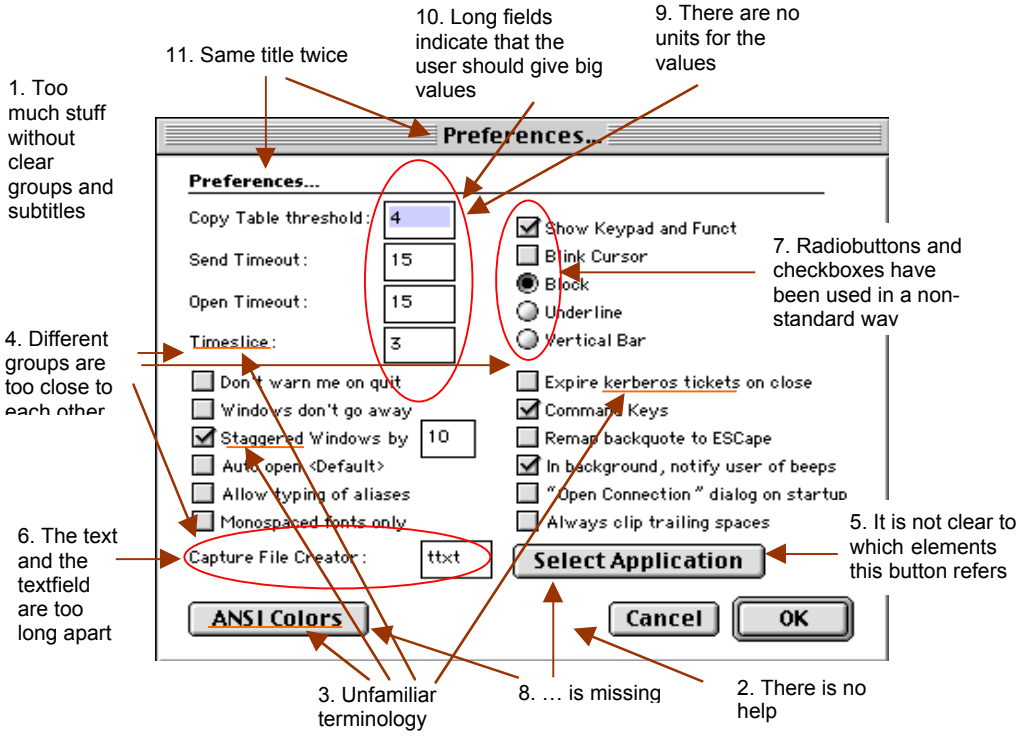

Figure 1: The results of a heuristic evaluation are easy to understand if they are clearly linked to the evaluated dialogue and its elements.

It is important to order the problems by their severity. This way, the designers can easily detect the most severe problems and fix them as soon as possible. Especially, if evaluators have detected many minor problems, major problems could vanish into the problem list.

Table 1: An example of the results of a heuristic evaluation. The evaluated dialogue is presented in Figure 1.

Problem no	The problem	Heuristics that the design violates	Severity 3: major 2: disturbs 1: minor
1	The dialogue contains too much stuff that is not in clear groups and does not have subtitles	Simple dialogue, Memory load	3
2	The dialogue does not provide help	Prevent errors, Help	3
3	The dialogue uses unfamiliar terminology	Users' language	3
4	The groups in the dialogue are too near to each other and therefore are hard to separate from each other	Simple dialogue	3
5	It is not clear to which group of elements the button Select Application belongs to	Simple dialogue, Prevent errors	2
6	The text Capture File Creator: and its text field are too long apart, so that they are hard to associate to each other	Simple dialogue, Prevent errors	2
7	Radio buttons and checkboxes are used in a same group	Simple dialogue	2
8	… is missing from the buttons ANSI Colors and Select Application although they open a new dialogue	Simple dialogue	2
9	There are no units for the values in the text fields	Memory load, Prevent errors	2
10	The text fields are quite long and thereby indicate that there should be big values	Prevent errors	1
11	The same title appears twice in the dialogue	Simple dialogue	1

1.2 Cognitive walkthrough

Our usability group has only a little experience with the cognitive walkthrough method. One reason for this is its reputation as a tedious method. Another reason is the fact that we usually get finished products to be evaluated and usability testing is more suitable for them.

Cognitive walkthrough is at its best with walk-up-and-use systems. Therefore, it is a good method for evaluating systems that need to be self guiding and easy to learn. We recommend a cognitive walkthrough for evaluating smart products, such as mobile phones, heart rate monitors and automatons for train tickets. The students have applied cognitive walkthrough also to software systems that they have designed and implemented as course assignments. The students have picked 3-5 core tasks of their system and analysed them from the viewpoint of presumed users. The walkthroughs have revealed, *e.g.*, missing controls, controls that seem more appealing than the right ones and terms that are not familiar to the users. Applying the method has been laborious, but the students have considered it worthwhile, because it has helped them

in finding problems so early in the development process that even big modifications have been possible.

1.2.1 Application to a drink can refund machine

To get a drink can returned, the user has to put the can into the round hole in the front panel of the machine. The can has to be in horizontal position, bottom first and the bar code on the top. The only indicator for the right position is a symbol of a bar code on top of the hole in the front panel. After setting the can into the hole, the user has to let the can go and wait for the machine to process the can. If the user keeps the can in his hands, the machine can not weigh the can and start the process. Three indicator lights indicate whether the can is accepted or not:

1. a red light indicates that the can is rejected
2. a yellow light indicates that the can is identified but it has no refund value
3. a green light indicates that the can is approved and the customer gets his deposit back.

Walking through users' tasks

Our team created scenarios of various use situations with the refund machine. We concentrated in error situations, because the guidance in these situations seemed bare and ambiguous. For instance, there were several reasons for the machine to turn on the red light:

- it weighed the can to be too heavy or too light
- it identified that the can was not of aluminium
- it did not find a bar code from the can, because the can was upside down or the bar code was not on the top
- it could not identify the bar code, because the code was damaged
- it could not find the code from its database of known bar codes.

Still, the user was supposed to know, how to fix the problem, or at least remove the can from the machine.

A drink can refund machine is a very typical walk-up-and-use system. Every customer should be able to use it instantly without any training or special guidance. Therefore, we applied cognitive walkthrough method to the first concept to evaluate typical error situations.

1.2.2 Ways to present the results

We have used the manual concept of the drink can refund machine as an assignment to learn and rehearse cognitive walkthrough in our courses. Usually, the students present the results as a list of questions and answers, but one group made a summary of its findings as a table. A part of this summary is shown in Table 2. This seemed to

be a good way to present a summary: the problems are easy to find. In addition to this summary, the results must include detailed and clear descriptions of the problems and an estimation of their severity.

Table 2: A table is a good way to present a summary of the results of cognitive walkthrough.

Right sequence of actions	Will the user try to achieve the right effect?	Will the user notice that the correct action is available?	Will the user associate the correct action with his goal?	Will he see that the task proceeds?
1. Put the can in the machine	OK, user wants to get rid of the can	OK	OK	OK, a light is lit
a) horizontally	OK, the hole is of right size	OK, the hole if of right shape	OK	OK, -"-
b) bottom first	NO, the user is not interested in the direction	NO	NO	NO, he does not know, why the can is rejected
c) code bar on the top	NO, the user is not interested in the position of the code	NO, bar codes are unfamiliar to some users	NOT necessarily	NO, -"-
2. Identify the feedback of the machine	OK, if the can is in right position; NO, if not	NO, does not possibly know the meaning of the lights	Maybe, but just by guessing	NO, the user does not know, how he should fix the problem

With some products, such as a heart rate monitor, we have noticed that it is easier to combine questions "Will the user notice that the correct action is available?" and "Will the user associate the correct action with his goal?" into one question: "Will the user find the correct action?" This includes the part of perceiving the action and the part of realising that the action is the one that is needed. The latter part means that the user interface uses terms and icons that are familiar to the user, or are easy to understand and to associate to the action. Table 3 shows a summary of a cognitive walkthrough for a heart rate monitor. The analysed task was starting the measurement of user's heart rate.

Table 3: A summary of a cognitive walkthrough for measuring heart rate with a heart rate monitor.

Right sequence of actions	Will the user try to achieve the right effect?	Will the user find the correct action?	Will he see that the task proceeds?
1. Put on the receiver	OK	OK	OK
2. Put on the transmitter	NOT necessarily: beginners might think that the receiver is enough	NOT necessarily: the transmitter has such a good place in the package that the user might not notice it	NO: the receiver does not respond if the transmitter is not wet
3. Moisten the transmitter	NO: a beginner does not know that the transmitter must be wet	OK: the user probably finds a way to moisten the transmitter	OK?, with a small delay