DISCOUNT USABILITY TESTING

By **Erik van Veenendaal**
Improve IT Services BV, Bonaire

Usability is an important aspect of software products. However, in practice not much attention is given to this issue during testing. Testers often do not have the knowledge, skills and/or time available to test usability. This paper identifies a number of easy-to-use and therefore low cost usability test techniques as a possible solution to this problem. The techniques identified, and briefly described, are heuristic evaluation, cognitive walkthrough and questionnaires. Heuristic evaluation involves having a small set of evaluators examine the interface and judge it's compliance with recognized usability principles ("the heuristics"). The cognitive walkthrough is an usability test technique in which one or more evaluators work through a series of tasks and ask a set of questions from the perspective of the user. Two example of usability questionnaires are discussed in this paper: SUS and SUMI. Both SUS and SUMI are rigorously tested and validated questionnaires to measure software quality from a user perspective. The intend of this paper is not to describe a number of usability test techniques in detail, but rather to point the reader in the direction of highly interesting easy-to-use, easy-to-learn, low cost usability test techniques. The author has applied the various techniques successfully many times in various projects and for many different types of software products.

A closer look at usability

Several studies have shown that in addition to functionality and reliability, usability is a very important contributor the success of a software product. Although it is possible to test the software extensively in a usability lab environment, in most situations an usability test has to be carried out with minimum resources (if at all).

The usability of a product can be tested from two different perspectives "ease-of-use" and "quality-in-use". Quite often the scope during usability testing is limited to the first perspective. The ease or comfort during usage is mainly determined by characteristics of the software product itself, such as the user-interface. In this context usability is a part of the product quality characteristics. The usability definition of ISO 9126 is an example of this type of perspective:

> *Usability (ISO 9126)*
>
> the **capability of the software** to be understood, learned, used and liked by the user, when used under specified condition

Types of techniques that can be used to test the usability aspect of product quality are heuristic evaluation and cognitive walkthrough. However, these techniques have the disadvantage that the real stakeholder, e.g., the user, isn't involved. In a broader scope usability is being determined by using the product in its (operational) environment. The type of users, the tasks to be carried out, physical and social aspects that can be related to the usage of the software products are taken into account. Usability is being defined as "quality-in-use". The usability definition of ISO 9241 is an example of this type of perspective:

> *Usability (ISO 9241)*
>
> the extent to which a product can be **used by specified users** to achieve goals with effectiveness, efficiency and satisfaction in a **specified context of use**

Clearly these two perspectives of usability are not independent. Achieving "quality-in-use" is dependent on meeting criteria for the usability aspect of product quality. This relationship is shown in figure 1.

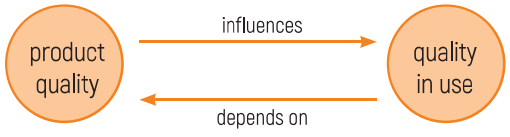

Figure 1: Relationship between different types of usability

Establishing usability test scenarios, for example based on use cases, can be used to test usability in accordance with ISO 9241. However, usability testing with specific usability test cases / scenarios is a step too far for most organizations. From a situation where usability is often not tested at all, but yet an important aspect of product quality, one wants techniques that are easy-to-use, reliable, and require limited resources. It almost sounds too good to be true, but there are some usability test techniques available that meet these criteria. In this paper subsequently heuristic evaluation, cognitive walkthrough and usability questionnaires will be briefly discussed.

Heuristic evaluation

Heuristic evaluation is a systematic inspection of a user interface or user interface design for usability. This technique takes the software inspection methodology and adapts it to usability evaluation. It's a usability test technique that focuses on the product quality aspect of usability.

With this technique, the reviewers examine the interface and judge its compliance with recognized usability principles (the "heuristics"). The goal of a heuristic evaluation is to find usability problems in the design so that they can be dealt with as part of an iterative design process. In general, heuristic evaluation is difficult for a single individual to do because one person will never be able to find all the usability problems in an interface. Experience from many different projects has shown that different people find different usability problems. Therefore, it is possible to improve the effectiveness of the method significantly by involving multiple evaluators. The effectiveness increases when evaluators are assigned specific roles based on the heuristics.

Jacob Nielsen, the founder of this technique, promotes a (popular) set of ten design heuristics summarized below. Note that also other sets of heuristics exist, and several collections of supporting checklists are available.

1. Visibility of system status
2. Match between system and the real world
3. User control and freedom
4. Consistency and standards
5. Error prevention
6. Recognition rather than recall
7. Flexibility and efficiency of use
8. Aesthetic and minimalist design
9. Help users recognize, diagnose, and recover from errors
10. Help and documentation

Figure 2: Ten Usability Heuristics defined by Jacob Nielsen

Heuristic evaluation is performed by having each individual evaluator inspect the interface alone. Only after all evaluations have been completed, are the evaluators allowed to communicate and have their findings aggregated. This procedure is important in order to ensure independent and unbiased evaluations from each evaluator. Typically, a heuristic evaluation session for an individual evaluator lasts one or two hours. Longer evaluation sessions might be necessary for larger or very complicated interfaces with a substantial number of dialogue elements, but it would be better to split up the evaluation into several smaller sessions, each concentrating on a part of the interface. During the evaluation session, the evaluator goes through the interface several times and inspects the various dialogue elements and compares them with the heuristics. These heuristics are general rules that describe common properties of usable interfaces. In addition to the checklist of general heuristics to be considered, the evaluator obviously is also allowed to consider any additional usability principles that come to mind and are relevant for the specific dialogue under evaluation.

In principle, the evaluators decide on their own how they want to proceed with evaluating the interface. A general recommendation would be that they go through the interface at least twice, however. The first pass would be intended to get a feel for the flow of the interaction and the general scope of the system. The second pass then allows the evaluator to focus on specific interface elements while knowing how they fit into the larger picture. Since the evaluators are not using the system as such (to perform a real task), it is possible to perform heuristic evaluation of user interfaces that exist on paper only and have not yet been implemented. This makes heuristic evaluation suited for use early in the development lifecycle. Heuristic evaluation is explicitly intended as a "discount usability engineering" method. Practical experiences have indeed confirmed that heuristic evaluation is both a very efficient and highly effective usability test technique.

Cognitive walkthrough

Cognitive walkthrough is a review technique where usability evaluators construct task scenarios from a specification or early prototype. They then role-play the part of a user working with that interface, by "walking through" the interface, working through typical tasks. They act as if the interface was actually built. Each step the user would take is examined. Where the evaluator is blocked by the interface and cannot complete the task, this indicates that the interface is missing something. If the path through the task is convoluted and circuitous this indicates that the interface and the software need simplifying.

The analysis phase consists of examining each step within a task and attempting to tell a credible story as to why the expected users would choose the correct action. Credible stories are based on assumptions about the user's background knowledge and goals, and on an

understanding of the problem-solving process that enables a user to guess the correct action.

As the walkthrough proceeds, the evaluators ask the following four questions:

1. *Will the users try to achieve the right effect?*

For example, their task is to print a document, but the first thing they have to do is select a printer. Will they know that they should select a printer?

2. *Will the user notice that the correct action is available?*

This relates to the visibility and understandability of actions in the interface.

3. *Will the user associate the correct action with the effect to be achieved?*

Users often use the "label-following" strategy, which leads them to select an action if the label for that action matches the task description.

4. *If the correct action is performed, will the user see that progress is being made toward solution of the task?*

This is to check the system feedback after the user executes the action.

The evaluator(s) will try to construct a success story for each step in the task. When a success story cannot be told, a failure story will be constructed, providing the criterion (one or more of the four questions above) and the reason why the user may fail.

The cognitive walkthrough technique is also largely focused on the product quality aspect of usability. However, since it also takes user tasks and the user's thinking process into account, it clearly has a stronger relationship to quality-in-use than the heuristic evaluation.

Questionnaires

In addition to, or instead of observation techniques, it is possible to use survey techniques and attitude questionnaires both during testing and once the product is used live. A user survey is a usability evaluation where users are asked to report subjective data into a questionnaire based on their experience in using a software product. User surveys can be used to evaluate the levels of user satisfaction with a software product. As such a usability questionnaire is related to the quality-in-use aspect of usability. The surveys may be "home grown", but in practice their development has often shown to be much more difficult than expected.

It is also possible, and indeed highly recommended, to measure against a benchmark, by using standardized questionnaires such as SUS (System Usability Scale) and SUMI (Software Usability Measurement Inventory). Both SUS and SUMI permit benchmarking against a database of previous usability measurements. They also provide concrete measurements of usability that can be used as completion or acceptance criteria.

System Usability Scale (SUS)

The SUS was invented by John Brooke, who created this 'quick and dirty' usability scale to evaluate practically any kind of system. The SUS has been tried and tested throughout many years of use, and has proven itself to be a dependable method of evaluating the usability of systems. The System Usability Scale is one of the most efficient ways of gathering statistically valid data and giving the software product a clear and reasonably precise score. It's basically a short quiz that doesn't require a lot of resources to administer, so very useful when one is constrained by budget but still needs good information fast. Despite being cheap and fast, the SUS is still valid – it measures what it sets out to measure and has shown to be solid and dependable.

The SUS is a Likert scale (see below) which includes 10 questions which users of your software product will answer. Participants will rank each question from 1 to 5 based on how much they agree with the statement they are reading. 5 means they agree completely, 1 means they disagree vehemently. Below are the 10 template questions that one can of course slightly adapt to better suit the context of a specific product:

1. I think that I would like to use this system frequently.
2. I found the system unnecessarily complex.
3. I thought the system was easy to use.
4. I think that I would need the support of a technical person to be able to use this system.
5. I found the various functions in this system were well integrated.
6. I thought there was too much inconsistency in this system.
7. I would imagine that most people would learn to use this system very quickly.
8. I found the system very cumbersome to use.
9. I felt very confident using the system.
10. I needed to learn a lot of things before I could get going with this system.

ERIK VAN VEENENDAAL
Improve IT Services BV, Bonaire

Erik van Veenendaal www.erikvanveenendaal.nl is a leading international consultant and trainer, and a recognized expert in the area of software testing and requirement engineering. He is the author of a number of books and papers within the profession, one of the core developers of the TMap testing methodology, a participant in working parties of the International Requirements Engineering Board (IREB), and currently a board member of the TMMi Foundation. Erik is a frequent keynote and tutorial speaker at international testing and quality conferences. For his major contribution to the field of testing, Erik received the **European Testing Excellence Award** (2007) and the **ISTQB International Testing Excellence Award** (2015). You can follow Erik on twitter via @ErikvVeenendaal.

This means that each question placed next to the Likert scale will look as below:

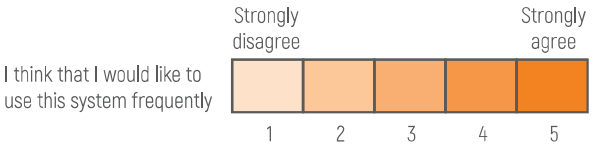

Figure 3: Sample statement from SUS

The SUS is not diagnostic and will not tell what the specific problems are, but it will provide a red or green light to know how badly usability needs work. The System Usability Scale and the tests above will assist in getting a definite grade for the usability of the software product. Rather than guessing, you'll know exactly how well you are doing benchmarked against industry standards.

Software Usability Measurement Inventory (SUMI)

SUMI is another solution to the recurring problem of measuring users' perception of the usability of software. It consists of a 50-item questionnaire devised in accordance with psychometric practice. Each of the questions is answered with "agree", "undecided" or "disagree". The following examples show the kind of questions that are asked:

- This software responds too slowly to inputs
- I would recommend this software to my colleagues
- The instructions and prompts are helpful
- I sometimes wonder if I am using the right command
- Working with this software is satisfactory
- The way that system information is presented is clear and understandable
- I think this software is consistent.

In order to use SUMI effectively a minimum of ten users is recommended. Based on the answers given and statistical concepts the usability scores are being calculated. Basically any kind of software product can be evaluated using SUMI as long as it has user input through keyboard or pointing device, display on screen, and some input and output between secondary memory and peripheral devices. When evaluating a product or series of products using SUMI (or SUS), one may either do a product-against-product comparison, or compare each product against the standardization database, to see how the product that is being rated compares against an average state-of-the-market profile.

Whereas SUS only give one final usability score, SUMI gives a global usability figure and then ratings on five subscales:

- *Efficiency:* degree to which the user can achieve the goals of his interaction with the product in a direct and timely manner
- *Affect:* how much the product captures the user's emotional responses
- *Helpfulness:* extent to which the product seems to assist the user
- *Control:* degree to which the user feels he, and not the product, is setting the pace
- *Learnability:* ease with which a user can get started and learn new features of the product.

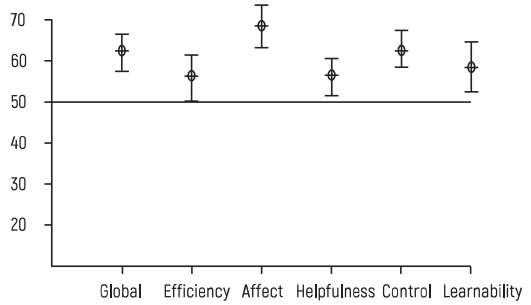

Figure 4: A sample profile showing SUMI scales

Figure 4 shows an example of SUMI output. It shows the scores of a test and the standard deviations of these scores against the average score of the reference database, which is reflected by the value 50. Consequently the usability scores shown in the sample profile are positive, e.g., more than state-of-the-art, and also within a reasonable level of standard deviation.

SUMI is a questionnaire for the assessment of usability of software, which has been developed, validated and standardized on a European wide basis. The SUMI subscales are being referenced in international ISO standards on usability and software product quality. Product evaluation with SUMI provides a clear and objective measurement of users' view of the suitability of software for their tasks.

Final comments

It has been said that a system's end users are the experts in using the system to achieve goals and that their voices should be listened to when that system is being evaluated. SUMI and SUS do precisely that: it allows quantification of the end users' experience with the software. It basically tells you whether you have a problem or not. Evaluation by usability professionals is also important, using techniques as heuristic evaluation and cognitive walkthrough has the advantage over SUMI and SUS that they indicate specific problems regarding usability. Ideally a questionnaire is blended with a usability evaluation.

There is much more to say about the usability test techniques described above, e.g., share some practical experiences and lessons learned. However, if you are a tester involved in a project or organization where usability is an issue but again there is just very little time, here are the techniques to consider. If you are interested, I urge you to start reading and digging into these techniques and prepare yourself since you may well need them tomorrow.