Zuerst ersch. in: Behaviour & Information Technology ; 12 (1993), 3. - S. 137-148

Evaluation of user interfaces: EVADIS II—a comprehensive evaluation approach

HARALD REITERER[1,2] and REINHARD OPPERMANN[1]

[1]Institute for Applied Information Technology (FIT), Human–Computer Interaction Research Division (HCI), German National Research Center for Computer Science (GMD), Schloß Birlinghoven, PO Box 1316, D-5205 St Augustin 1, Germany
email reiterer@gmdzi.gmd.de or oppi@gmdzi.gmd.de
[2]Institute for Applied Computer Science and Information Systems, University of Vienna, Liebigg. 4/3-4, A-1010 Vienna, Austria

Keywords: Usability; system evaluation; user interface.

Abstract. As a result of the importance of the usability approach in system development and the EC's 'Directive concerning the minimum safety and health requirements for VDT workers' (EWG 1990), there is an accepted need for practical evaluation methods for user interfaces. The usability approach and the EC Directive are not restricted to user interface design, as they include the design of appropriate hardware and software, as well as organization, job, and task design. Therefore system designers are faced with many, often conflicting, requirements and need to address the question, 'How can usability requirements comprehensively be considered and evaluated in system development?' Customers buying hardware and software and introducing them into their organization ask, 'How can I select easy-to-use hardware and software?' Both designers and customers need an evaluation procedure that covers all the organizational, user, hard- and software requirements. The evaluation method, EVADIS II, we present in this paper overcomes characteristic deficiencies of previous evaluation methods. In particular, it takes the tasks, the user, and the organizational context into consideration during the evaluation process, and provides computer support for the use of the evaluation procedure.

1. Why is there an increasing need for effective evaluation methods?

Workplaces today are equipped commonly with visual display terminals (VDTs); more and more workers use interactive applications. There are estimates that by the year 2000 about 90% of *all* employees in industrialized countries will utilize VDT of one kind or another (Fähnrich 1987).

When computer applications are developed for the workplace, technical questions are often overemphasized, to the neglect of organizational and social impacts. This often results in hard-to-use, user-unfriendly applications. The consequences for the employee are frustration, anxiety, and stress; the consequences for the company are decreased organizational flexibility, absenteeism, staff turnover, and thus performance decrement (Greutmann 1992). Against this background, application characteristics like 'user-friendliness', 'ease-of-use', 'usability', or 'ergonomic design' have been recognized as essential. The International Organization for Standardization (ISO 9241 Part 1) defines good ergonomic design of VDT work as '. . . to ensure that VDT users can operate display screen equipment safely, efficiently, effectively, and comfortably. In practice, this can only be achieved by careful design of the VDTs themselves, the workplaces, and working environments in which they are used and the way the VDT work is designed, organized and managed'. This definition is not restricted to user interface design; it includes the design of appropriate application functionality, organizational design, and job and task design. Therefore system designers are faced with many—often conflicting—requirements, and need to address the question, 'How can usability requirements be considered and evaluated during application development?' To consider usability requirements, designers need design criteria and design rules, appropriate design methods, and design tools. To evaluate usability during and after the development process, they need appropriate evaluation methods that provide feedback on the ergonomic quality of their work. In these terms, usability is an integral part of

software quality in general. An example of such an evaluation method, EVADIS II, is presented in this article.

Another reason for the increasing need for evaluation methods is the new European Economic Area (EEA, consisting of EC and EFTA). To establish common working conditions for VDT users, the European Community published a 'Directive concerning the minimum safety and health requirements for VDT workers' (EWG 1990); national governments of the EC have been required to enshrine this Directive in national law. In this process the European standardization activities of the CEN (Comité Européen de Normalisation) and the international standardization activities of the ISO concerning ergonomic requirements for VDTs have had significant influence, especially ISO standard 9241 'Ergonomic requirements for office work with VDTs' (CEN 29241) (Cakir 1991). In the future this standard may be an integral part of software requirements specification. Software developers will have to take its requirements and principles into consideration, including conformance testing products against the standard. On the other hand, software buyers also need evaluation methods to test conformance with the standards. So both groups, developers and buyers, need effective, practical software evaluation methods.

2. Which factors influence an evaluation?

Whitefield *et al.* (1991) define human factors evaluation thus: 'Human factors evaluation is an assessment of the conformity between a system's performance and its desired performance.' System performance is a system's effectiveness in accomplishing tasks. One must consider the quality of the task product (i.e., how well the task's outcome meets its goal) and the incurred resource costs (i.e., the resources employed by both the user and the computer in accomplishing the task). The desired performance is determined by usability goals and ergonomic design principles. The term assessment involves both a method (the process by which it is done) and a statement (the resulting product). The term 'system' means in an ergonomic sense a *user* and a *computer* (hardware and software) engaged upon some *task* within an *organization*. So a complete evaluation of human–computer interaction must consider the user, the tasks, the computer, and the organization. Figure 1 shows these factors and the relationship between them (Frese and Brodbeck 1989). The relationship 'accomplish tasks' describes how a user can carry out the tasks. The relationship 'usability' describes how easy/difficult it is for the user to use the software. The relationship

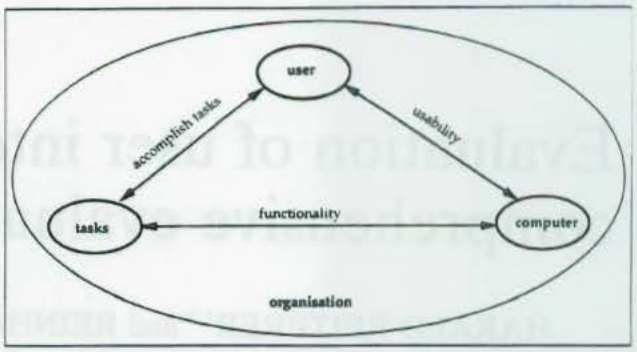

Figure 1. Factors to be considered during the evaluation of user interfaces.

'functionality' describes how well/badly the software supports the tasks and allows the user to reach the task goals. An evaluation that takes all these factors and relationships into consideration could be called holistic or comprehensive.

3. Which evaluation methods can be used?

3.1. *What evaluation methods are available?*

Today many evaluation methods are available but no one is sufficient alone. Each method has its advantages and disadvantages, as the following classification shows.

3.1.1. *Subjective evaluation methods*: Subjective evaluation methods are directly based on the user's judgement. The user is the source of the evaluation, possibly even its initiator. The user of a system is asked questions about certain system properties. The answers are based on his or her accumulated experience. A distinction must be made between oral and written questionnaires. Another method is 'thinking aloud' whereby users perform a task while giving verbalizing their thoughts, problems, opinions, etc., all of which enables the evaluator to interpret the test. As this approach may seem artificial to the user, an alternative is the 'constructive interaction' method, in which two users work together on a task and 'tell' each other what they are feeling, doing, or intending to do, etc. This generates data in a more 'natural' manner. Subjective evaluation methods tend to yield subjective ('soft') data (e.g., whether the system is comfortable, easy to use, manageable, comprehensible, etc.) rather than objective ('hard') data (e.g., whether a system performs a task quickly; whether it is error free). The advantages of subjective methods are those of low cost; ease of implementation; and an ability to pin-point unstructured problems, etc. The drawbacks are a tendency to

produce exaggerations; the difficulty in avoiding leading questions; a plethora of data, which makes evaluation a costly matter; and the low regard in which such methods are held by those questioned. Examples of subjective evaluation methods based on written questions and answers that can be practically applied are the Questionnaire for User Interaction Satisfaction (QUIS 5.0) developed by Norman and Shneiderman (1989) or the Evaluation Checklist, developed by Ravden and Johnson (1989). A new and interesting subjective evaluation approach is the Software Usability Measurement Inventory (SUMI) of the MUSiC Project (MUSiC 1992).

3.1.2. *Objective evaluation methods*: Within objective evaluation procedures, we find a large number of approaches, ranging from para-experimental studies (e.g., 'Wizard of Oz'), through the evaluation of system properties on the basis of checklists, to classical experiments. The advantage of objective evaluation methods is that they are not based on subjective judgements by users or evaluators (they avoid 'soft' data). The disadvantage of objective evaluation methods is their limited scope of observation (they produce only 'hard' data). For example, in logfiles of user behaviour with input media, recorded by the computer itself, observation may be concealed, but is necessarily confined to the user's handling of the system. Any other actions or interactions, e.g., signs and gestures, exclamations, use of manuals, communication with others, etc., are not recorded. A good example of an objective evaluation method is the Performance Measurement Method for Usability Evaluation of the MUSiC Project (MUSiC 1992).

3.1.3. *Guideline-oriented evaluation methods*: These methods lie at an intermediate stage between subjective and objective evaluation methods. In these methods, a system is examined by an expert. Unlike the question-and-answer sessions discussed earlier, the expert's approach derives less from a task to be performed by the tested system, and more from questions prompted by software ergonomics. These methods are subjective since the expert examines and answers questions according to her or his personal assessment. They are objective since the examination criteria of software ergonomics are operationalized and precisely formulated to an extent enabling the evaluator to answer questions on the basis of clear test rules and traceable conditions. The advantages are that the guideline-oriented evaluation method (expert judgement) is relatively fast, uses few resources, provides an integrated view, and can address a wide range of behaviour. On the other hand, its reliability will vary between experts, and since its assessments are inevitably subjective, its reports are likely to be incomplete, biased, and difficult to validate (Whitefield *et al.* 1991). Detailed instructions in the evaluation guide (e.g., detailed process description, clear notation, structure of the statement) can help reduce the subjectivity of this method. Hammond *et al.* (1985) report a comparison between expert judgement and user observation and demonstrate expert judgement to be superior.

There are now a number of guideline-oriented checklists for experts. For example, the checklist of the Bavarian testing authority, TÜV Bayern (Lang and Peters 1988); or the extensive compilation of questionnaires for evaluating the use of new technologies in a company (Clegg *et al.* 1988). An important measure for guideline-oriented evaluation methods is the extent to which they are embedded in a test scheme, i.e., in a test specification for the performance of an evaluation. Many allow the evaluator to specify the way the system under test should be used in order to obtain answers to test questions. In addition to the test questions proper, some guideline-oriented methods also specify the evaluation procedure, e.g., the system EVADIS II described herein.

3.1.4. *Experimental evaluation methods*: Among experimental evaluation procedures, 'benchmark tests' play an important role. These involve comparing the way different systems perform certain standardized tasks. A case in point is the study by Roberts and Moran (1983) involving nine text editors. Benchmark tests do not yield absolute statements about systems, but involve placing different systems on an ordered scale on the basis of defined criteria. The comparative nature of benchmark tests does not necessarily apply to other experiments, e.g., experiments testing theories. Well-known examples in this respect are experiments testing Card *et al.*'s (1983) GOMS model.

One problem involved in planning experiments is the correct definition of dependent and independent variables. A second problem is the selection of the proper environment for the study. A third problem is the lack of any underlying theory dealing with human–machine interaction, so that the features to be considered are often left to the researcher's imagination and sympathies.

3.1.5. *Classification form*: A useful classification schema, in some cases similar to the former, is presented by Whitefield *et al.* (1991), where again four groups of evaluation methods are distinguished:

- analytic methods (e.g., GOMS-Model, CCT, TAG);
- specialist reports (e.g., expert judgement);
- user reports (e.g., questionnaires, interviews, rating methods);

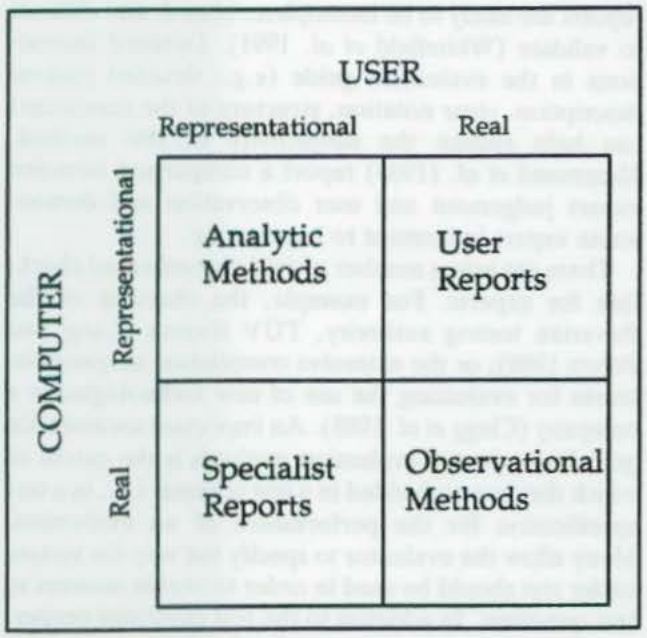

Figure 2. Classes of evaluation methods (Whitefield *et al.* 1991).

- observational methods (e.g., informal observation, full-scale experimentation).

The four methods are attached to real or representational computers and users as shown in figure 2. 'Real computer' means the physical presence of a computer. Thus implemented systems, prototypes, and simulations all count as real computers. On the other hand, specifications and notational models are representational computer presences, as are users' mental representations of the computer. 'Real user' means actual users or approximations of them (for example, students). In contrast, the presence of representational users means user descriptions or user models.

3.2. *A combination of evaluation methods*

Based on the requirements of a comprehensive evaluation described in section 2, and on the above classifications of methods, we have analysed the following evaluation methods: Baitsch *et al.* (1989), Clegg *et al.* (1988), ETH-LAO (1986), Hoyos *et al.* (1990), Lang and Peters (1988), MITRE (1986, 1991), Norman and Shneiderman (1989), Oppermann *et al.* (1989), Ravden and Johnson (1989), Sherwood-Smith (1989), Siemens (1987, 1990), Simes and Sirsky (1985), TBS (1991), and Tullis (1988).

Reiterer (1990) shows that there is no single 'best' evaluation method. All of the methods examined have some disadvantages, or consider only a limited number of the factors influencing an evaluation, but many of them contain useful ideas, or are very appropriate for

methods / focus of the evaluation	interview techniques	simplified task analysis	expert judgement
user	aim: to explore user characteristics tool: standardized questionnaire		
tasks and organization		aim: to examine and evaluate the tasks and working conditions; to construct a test task tool: guideline	
software			aim: to assess the usability of the user interface tool: evaluation software EVADIS II (list of test items)

Figure 3. Combination of methods for a comprehensive evaluation.

the evaluation of a specific factor. What is needed is a combination of different evaluation methods for the different foci of an evaluation (see also Piepenburg and Rödiger 1989, Kishi and Kinoe 1991). Figure 3 exemplifies a combination of methods. For each factor—as a focus of the evaluation—a specific method is chosen. No hardware evaluation method is described because there are many useful guidelines and checklists for this purpose (for example, Köchling 1990, Grandjean 1987).

In order to explore the characteristics of the typical *users* of a piece of software, interview techniques can be used. It is good practice to use a standardized questionnaire to reduce the amount of time (subjective method/user report). The *tasks*, typically supported by the software and the user's working conditions, can be examined and evaluated by the use of task analysis methods (objective method/observational method). Well known task analysis methods such as VERA/B (Rödiger *et al.* 1986), KABA (Dunckel 1989, Zölch and Dunckel 1991) or TBS-GA (Rudolph *et al.* 1987) for office work, are available, but they are all very expensive and need comprehensive knowledge of ergonomics. For the purpose of evaluating the user interface it is enough to use a simplified task analysis in form of a simple guideline (for example VBBA, Bonitz *et al.* 1988, Bonitz 1989). To assess the usability of the user interface of the *software* an expert judgement is particularly useful (specialist report).

3.3. *How can we apply evaluation methods to the system development process?*

The software development process is typically structured in phases that specify the typical activities of the designer during the development process. There are a large number of different software development life cycle models. All life cycle models are divided into phases—for example (Olle *et al.* 1988):

- planning;
- analysis;
- design;
- construction;
- installation and test;
- operation and maintenance.

The goal of dividing the development process into well-defined phases is to ensure better project planning and sustained control of progress in development. The sequential structure of this life cycle model has often been criticized, especially with respect to usability issues. Iterative or evolutionary approaches have been proposed as alternatives. These often rely on software prototyping, which tries to integrate changing requirements due to user feedback into the development process (Floyd 1984).

An important point is the timing of an evaluation in the development process of a system. Timing affects development costs, because the costs of design modification are higher during later stages of development. It is clear that the evaluation should be an integral part of the whole development process, from the beginning. But many of the existing evaluation methods are very difficult to apply in an actual development process, because the costs of applying them appear too high. Thus system designers need criteria for understanding which evaluation methods are available and useful at different stages of the development process. Kishi and Kinoe (1991) present four criteria:

- The *time* an evaluation method can be conducted varies because some evaluation methods need real computers or users, while others can be used with representational computers or users.
- The *type and number of usability problems* which the evaluation method can detect depends on the class of usability problems the method is designed to address (e.g., hardware versus software, spatial design versus temporal design, application specific versus generic).
- The *workload* imposed by an evaluation depends on the time, the number of people required, on the knowledge necessary, etc., and varies in accordance with the method used.
- *Variations in measurement* caused by the evaluators are not welcome if a design decision must be based on reliable data (e.g. methods which rely on subjective judgement of evaluators or users).

In reality no existing evaluation method satisfies all these criteria simultaneously, because there are differences between the nature of a development process and that of a usability evaluation process. Development is more a top-down process, which goes through various stages from functional specifications to implementation, whereas a usability evaluation is more a bottom-up process. This means that something has to exist before one can use it in a real context, and then evaluate it. In practice one needs a combination of different evaluation methods, which complement each other and can be used at appropriate stages of the development process. To reduce the gap between the nature of the development and that of the evaluation process, early or rapid prototyping is a useful system development method. The prototyping approach should be combined with an evolutionary development process. This allows consideration of the results of the evaluation in the development (Eason 1982, Mambrey *et al.* 1986). The design and evaluation process has to be integrated into the system development process (Sherwood-Smith 1989).

The proposed combination of evaluation methods (see section 3.2) could be integrated into any life cycle model of the development process. The exploration of the user characteristics and the examination of the tasks and the organization can be achieved during the specification of system requirements in the *analysis* phase. It is one of the aims of this phase to analyse the users and their tasks, so as to obtain the necessary information for the evaluation as a by-product of the development process. If a prototyping approach is used, the assessment of the usability of the user interface can be done after the development of the first prototype in the phase *design*. The results can be taken into consideration during the evolutionary development of further prototypes. If no prototyping approach is used, the assessment of the user interface could be placed in the *installation and test* phase as part of the final quality control of the software.

For the evaluation of standard software, the life cycle model has planning and analysis phases, which are needed to specify the necessary requirements and to plan the introduction process, but no design and construction phases. These are instead replaced by a phase named *selection of standard software* (Koch *et al.* 1991). The assessment of the user interface with the help of an evaluation method could be one important aspect of the decision-making process for choosing standard software.

4. EVADIS II: a new evaluation approach

EVADIS II[1] was developed at the Human–Computer Interaction Research Division at the German National Research Center for Computer

[1] EVADIS II is based on EVADIS I which was developed from 1985 to 1988 by the Man–Machine Communication research group at the GMD (Oppermann *et al.* 1989).

Science (GMD), in close co-operation with the Institute for Applied Computer Science and Information Systems at the University of Vienna. The final version of the evaluation procedure was published in Oppermann *et al.* (1992).

EVADIS II is based on the combination of methods described in section 3.2, which is an initial step toward a comprehensive evaluation procedure. But it is clear that with this pragmatic combination of methods not all aspects of a comprehensive evaluation can be covered. The limitations of EVADIS II are described in section 4.6.

4.1. *Evaluation software*

EVADIS II provides computer support for the evaluation procedure. The evaluation software is implemented in Clipper™ and runs under DOS™ on an IBM-compatible PC. For the evaluation two computers are necessary: one for the software to be evaluated and one for the evaluation software (e.g., a laptop). The evaluation supporting software presents all test items on the screen in the sequence of the test task. The evaluator has to enter the answers, a rating, the explanation of her or his rating, and perhaps a note. After completing the test task the software supports the evaluator in assessing the user interface. It calculates an average mark for each ergonomic criteria and can sort the test items either by technical components or ergonomic criteria. So the evaluator is freed of routine work and can concentrate her or his activities on the evaluation process.

4.2. *Users of EVADIS II*

Typical users of EVADIS II might be developers and vendors of office software, organizations wanting to buy office software, management consultants, or trade unions. Because of its novelty we have no feedback from such organizations to date.

Another important area of use for EVADIS II is the field of education. Several German and Austrian universities (e.g. Vienna, Koblenz, Dresden, Berlin) use EVADIS II as a means of instruction for students of computer science, and all report good experiences with EVADIS II. Their students are able to learn the basic concepts of human factors and the use of evaluation methods in a playful way.

4.3. *The evaluation procedure of EVADIS II*

The EVADIS II evaluation procedure consists of the following five steps, which are described in detail in the EVADIS II evaluation guide:

1. installation and exploration of the software to be tested;
2. examination and evaluation of the tasks; selection of relevant test items; construction of test task(s);
3. exploration of user characteristics;
4. evaluation of the software using the test task(s);
5. interpretation of the results and drawing up of a test report.

Figure 4 gives an overview of the evaluation procedure and the EVADIS II components necessary to execute the five evaluation steps. The first three steps can be executed simultaneously. The result of these three steps is a test task, used as a 'script' to evaluate the software, and a ranked list of the ergonomic criteria. The order of this list reflects the importance of each ergonomic criterion for the particular user group. The intention is to ensure that the user characteristics are taken into consideration both during the evaluation of the user interface as well as during the interpretation of the test results. Step 4 is the central step of the evaluation process. Here all selected test items have to be answered. They are embedded in the test task. So the evaluation process alternates between the test task operations and answering of the associated test ques-

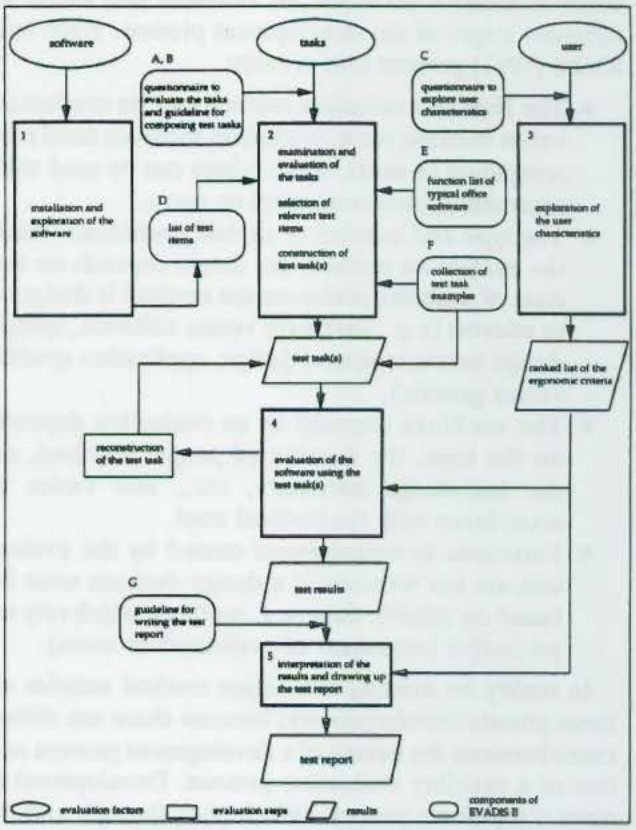

Figure 4. Evaluation procedure of EVADIS II.

tions. The result of these activities is a test record which forms the basis for the interpretation of the results and the writing of the test report.

4.4. *The components of EVADIS II*

The following components of the EVADIS II procedure are used during the evaluation procedure to instruct the evaluator (see figure 4):

(A) questionnaire to evaluate the tasks;
(B) guidelines for composing test task(s);
(C) questionnaire to explore user characteristics;
(D) list of test items;
(E) list of typical functions of office software;
(F) collection of test task examples;
(G) guidelines for writing the test report.

The description of the essential components of EVADIS II which follows is meant to familiarize the reader with the underlying concepts of this evaluation approach.

The questionnaire to evaluate the tasks (A) consists of 25 items which the evaluator has to answer during the examination of the tasks and working conditions (step 2). The questionnaire examines the characteristics of the existing tasks. The questionnaire is based on the suggested list of good task characteristics from ISO 9241 Part 2:

- Do the tasks need a variety of skills appropriate to the user's abilities?
- Are the tasks identifiable as whole units of work?
- Do the tasks provide the user with an appropriate degree of autonomy?
- Do the tasks provide the user with feedback on his performance?

The following example shows a typical item for evaluating the quality of the tasks:

Component 421: Distribution of work — Test item No.: 421.05.10
Criteria 05: Autonomy
Question: Can the employee decide in which manner, in which order and with which methods and tools he or she can accomplish his or her tasks?
Reference: Bonitz, 1989
Answer options: () no () sometimes () yes
Notes: _____

The interpretation of the results is simple because the answer options to the test items are limited. A simple rating form helps the evaluator to write the final statement about the ergonomic quality of the existing task. This statement will be included in the final test report (step 5). It is clear that with the help of such a short questionnaire it is impossible to detect all ergonomic limitations and deficiencies of the tasks at the workplace. That is not the aim of this questionnaire. It is only intended to give the evaluator a first impression of the ergonomic quality of the work and to show her or him serious deficiencies. If such deficiencies are detected, complete task analysis methods like VERA/B (Rödiger *et al.* 1986), KABA (Dunckel 1989, Zölch and Dunckel 1991) or TBS-GA (Rudolph *et al.* 1987) should be used for a detailed analysis.

The guideline for composing test task(s) (B) consists of detailed instructions and a collection of query-sheets (step 2). With the help of the guideline the evaluator has to examine the tasks that the users of the software are carrying out or plan to carry out. Using a combination of observation and questioning the user, the evaluator completes the query-sheets. These query-sheets include questions on the following topics:

- description of the working environment and the workplace where the software is used (organizational embedding);
- overview of the user's tasks at the workplace and determination of the tasks which are supported or will be supported by the software;
- description of the software supported tasks and a list of the associated software functions;
- overview of the hardware environment necessary for the software.

Next, before composing the test task(s), the evaluator has to read the complete list of test items (D), in order to select the relevant ones and include them in the test task. Based on the results of the observation and querying process, the evaluator is able to compose test task(s). The task analysis shows the evaluator which of the software functions are normally used to accomplish typical tasks. It also shows the importance of each function for this purpose. So the final test task is a 'script' consisting of all functions needed to accomplish one or more typical task(s) and the relevant test items. Depending on their content, the test items are placed after a sequence of test operations. The following example shows a small part of a test task.

...
...
9. Delete a file
Make a copy of the file "report" and delete the original file.
Test item no.: 241.07.20 Content of item: nonreversible action
Test item no.: 320.05.20 Content of item: deletion
10. Print a file
Print the file "report" on the local printer. Specify the layout of the document before printing (format, size, number of copy etc.).
Test item no.: 120.10.20 Content of item: choice of actions
Test item no.: 320.03.10 Content of item: functionality
Test item no.: 140.06.10 Content of item: output medium
...
...

EVADIS II includes a collection of test task examples (F), which guide the evaluator in this composition process. These examples show the typical structure of a test task and how the test items should be embedded in the test task(s).

The questionnaire to explore the user characteristics (C) is a collection of questions that the software users have to answer (step 3). The questionnaire includes 12 questions about user characteristics like knowledge and experience with hard- and software. The following example shows a representative question from this questionnaire:

Question: Which of the following do you feel you have some knowledge of, possibly from your education, or training courses, or having learned on your own? (You can mark more than one option)

Answer options:

a) basic knowledge

() how to operate the system

() about hardware and software

() about new tasks caused by the use of computers

b) extended knowledge

() about programming

() about operating systems

() about the design and use of data base systems

() about human factors

Notes: _____

An interpretation procedure and a classification of typical user groups guides the evaluator during the interpretation of the results. EVADIS II distinguishes between four different user groups: experienced and regular user, experienced and sporadic user, inexperienced and regular user, and inexperienced and sporadic user (Triebe *et al.* 1987). Each user group has an associated ranked list of software-ergonomic criteria, which shows the importance of each criteria (high, medium, low) for this user group. The ranked lists are based on a psychologic theory called 'control concept' (Spinas 1987). These ranking will be used for weighting the results of the evaluation in step 5: in the final assessment overview—generated by the evaluation software—the criteria are sorted in the order of their weighting (see section 4.5). It is clear that an important criteria should have a higher average rating than a less important criteria. Based on the weighting in comparison with the average rating the evaluator is able to formulate a differentiated usability assessment.

The list of test items (D) consists of about 150 items and is the core of EVADIS II. The items are used to evaluate the various properties of the user interface during the test task(s) (step 4).

The list is based on extensive studies of the available literature (especially standards, guidelines and style-guides), on the knowledge and experience of the authors, and on the assessment of existing evaluation procedures. To reduce the number of test items, logically-related ergonomic requirements are condensed into one item. The different requirements are presented with the help of a broad spectrum of answer options. The benefit of such consolidation is that the handling of the item list is much easier. Nevertheless the list of test items is only a representative selection of ergonomic requirements. It is up to the evaluator—based on the analysis of the tasks and the user group—to adapt these test items or to create new ones. The evaluation software offers some useful features for this purpose.

All items are embedded in a two-dimensional framework. Figure 5 shows this framework in some detail because it gives an important advantage over other evaluation methods. The first dimension is the technical system components, which distinguishes between four levels of the user interface: the input/output interface, the dialogue interface, the functional interface, and the organizational interface. These are basically inspired by the IFIP model for user interfaces (Dzida 1983). The second dimension is the software-ergonomic criteria. These are primarily based on the dialogue principles proposed by ISO 9241 Part 10 (CEN 29241) and by DIN 66234 Teil 8. The reason for using the ISO principles is their increasing importance for the background of the EC directive. EVADIS II includes four further criteria: 'availability' of hard- and software, 'clarity' of the presentation of information, the influence of the software on 'co-operation and communication', and the mechanism for 'data protection'. These four principles have been added because—in our opinion—they represent important ergonomic requirements that are outside the ISO principles.

Presenting the test items in this two-dimensional framework helps to explain to the evaluator the content of the various items and supports the search for the specific properties being investigated. It also ensures the completeness of the list of EVADIS test items. Figure 5 shows the two-dimensional framework and the location of some sample items.

Examples of test item questions shown in figure 5 are:

1. How is the user interface structured?
2. Is it possible to change the dialogue technique in different dialogue situations?
3. Can objects (e.g. documents) from one software module (e.g., word processing system) be copied into another software module (e.g., drawing program)?
4. Is it possible to exchange information with other users using the software?

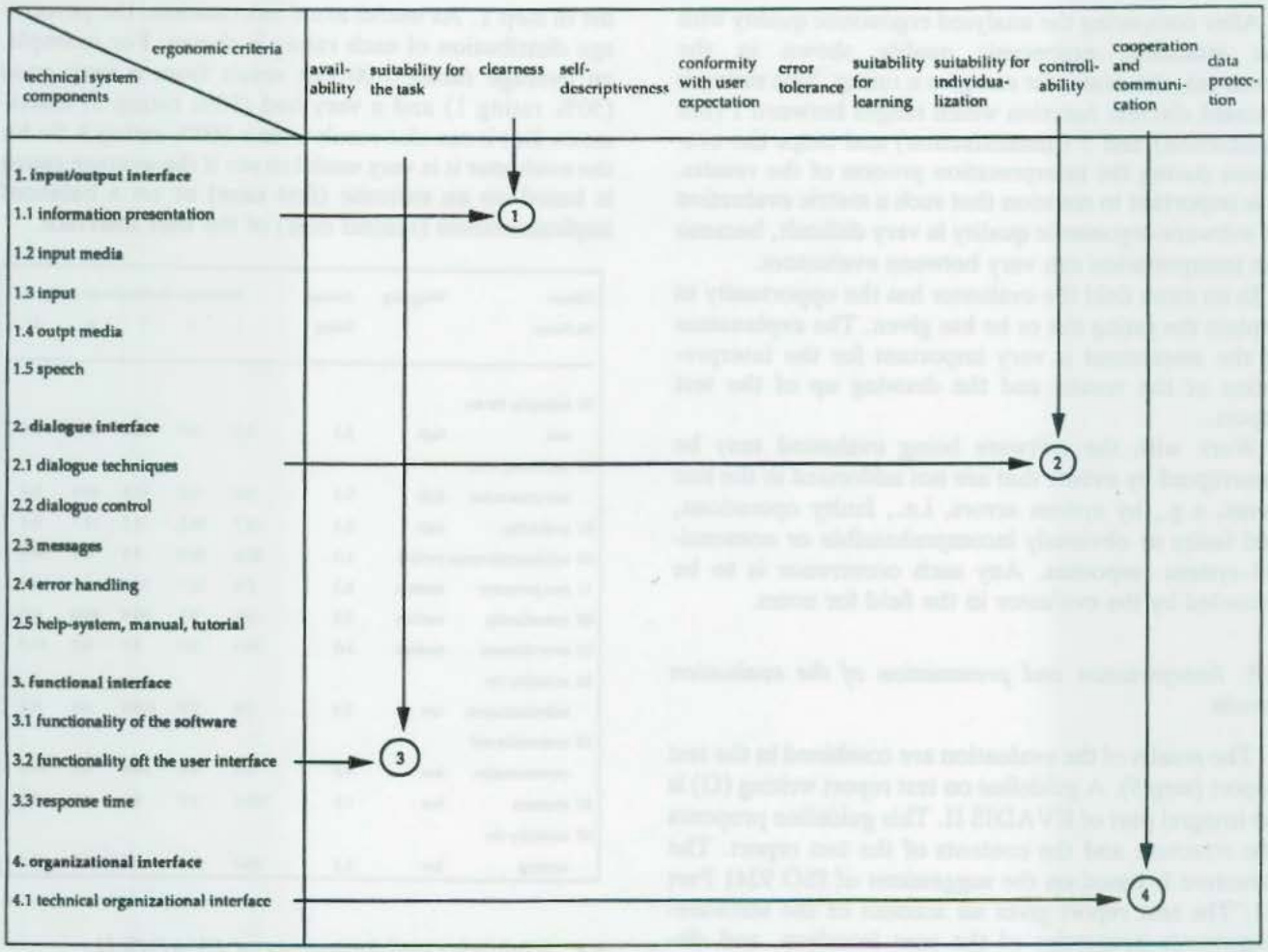

Figure 5. Two-dimensional framework and location of the test items.

Each test item includes a question, a set of possible answers, a comment, a field for a rating, a field for the explanation of the rating, and a field for notes.

Criteria: Self-descriptiveness **Test item No.:** 230.05.40

Component 230: Messages

Group 2: Check at the end of the test task, if necessary with the help of the manual.

Question: Does the system give status messages on current execution of background processes (processes handled independently of current dialogue)?

Answer options:

() yes, continuously

() yes, upon request

() can be switched off

() no

Comment: Status reports are desirable, but the user should be able to switch them off.

Assessment:

Rating: _____ **Weighting of the criteria:** _____

Explanation: _____

Notes: _____

The answer options cover the general spectrum of possible replies given the current state of R&D into software ergonomics. They help the evaluator to describe the ergonomic quality of the software. However, they are not complete: first, because technical developments or new knowledge may reveal new options that are not included here; second, because they provide only a rough guide for the evaluator in finding answers and checking functions. Any special feature not considered in the answer options must be noted separately by the evaluator.

Comments have been included with each item to help the evaluator interpret and assess the questions and answers. They are based on present knowledge of software ergonomics and indicate which of the answer options are considered currently to be the best interface features. Obviously, such comments are subject to rapid change. They are the best indicator of the progress made in software ergonomics and will be regularly updated.

After comparing the analysed ergonomic quality with the attainable ergonomic quality shown in the comment, the evaluator can give a rating. This rating is a scaled discrete function which ranges between 1 (full satisfaction) and 5 (dissatisfaction) and helps the evaluator during the interpretation process of the results. It is important to mention that such a metric evaluation of software-ergonomic quality is very difficult, because the interpretation can vary between evaluators.

In an extra field the evaluator has the opportunity to explain the rating she or he has given. The explanation of the assessment is very important for the interpretation of the results and the drawing up of the test report.

Work with the software being evaluated may be interrupted by events that are not addressed in the test items, e.g., by system errors, i.e., faulty operations, and faulty or obviously incomprehensible or nonsensical system responses. Any such occurrence is to be recorded by the evaluator in the field for notes.

4.5. *Interpretation and presentation of the evaluation results*

The results of the evaluation are combined in the test report (step 5). A guideline on test report writing (G) is an integral part of EVADIS II. This guideline proposes the structure and the contents of the test report. The structure is based on the suggestions of ISO 9241 Part 11. The test report gives an account of the software-ergonomic properties of the user interface, and discusses the answers to and ratings of the test items, including all additional test protocol notes. The evaluator's assessment is mainly based on the ratings and the explanations of the ratings but it can only be tentative, since sound software-ergonomic findings are not yet available for many interface properties. Interpretation also involves cross-references between different interface properties, e.g. regarding adherence to the principle of internal interface consistency.

The order in which answers are presented will vary from case to case, depending on the purpose. One account of software-ergonomic properties might be arranged by technical components. Such a format would particularly suit the needs of a designer who wants to know as precisely as possible where a trouble-spot is located. For other assessments, e.g., involving a decision on whether to buy a software product or not, a criterion-based report may be required.

The following is an example of an assessment overview that is generated automatically by the evaluation software, after the evaluation is finished. The overview shows the average rating for each criteria. The criteria are sorted by their weighting determined by the ranked list in step 1. As useful extra information, the percentage distribution of each rating is shown. For example, an average rating 3.00 can result from a very good (50% rating 1) and a very bad (50% rating 5) assessment. But it can also result from a 100% rating 3. So for the evaluator it is very useful to see if the average rating is based on an extreme (first case) or on a balanced implementation (second case) of the user interface.

Criteria No	Name	Weighting	Average Rating	Percentage-distribution per rating 1	2	3	4	5
02	suitability for the task	high	3.5	0.0	0.0	50.0	50.0	0.0
05	conformity with user expectation	high	3.6	0.0	0.0	40.0	60.0	0.0
01	availability	high	2.5	16.7	33.3	33.3	16.7	0.0
04	self-descriptiveness	medium	1.5	50.0	50.0	0.0	0.0	0.0
11	data protection	medium	2.3	0.0	66.7	33.3	0.0	0.0
09	controllability	medium	3.6	0.0	0.0	40.0	60.0	0.0
06	error tolerance	medium	3.0	50.0	0.0	0.0	0.0	50.0
08	suitability for individualization	low	3.0	0.0	0.0	100.0	0.0	0.0
10	cooperation and communication	low	4.0	0.0	0.0	50.0	0.0	50.0
03	clearness	low	1.0	100.0	0.0	0.0	0.0	0.0
07	suitability for learning	low	1.5	50.0	50.0	0.0	0.0	0.0

4.6. *Highlights and limitations of EVADIS II*

To describe the highlights and deficiencies of EVADIS II, we use the criteria described in section 3.3. EVADIS II needs a real computer and real users, so the timing of the evaluation in the development process could be after the stage of designing a prototype and having analysed the tasks and the user characteristics. Therefore EVADIS II cannot be used during the specification stage of the system development, where the system designer must use a prototyping approach. It is clear that EVADIS II can be used for post-evaluation purposes, like evaluating standard software products for purchase decisions. The primary focus of EVADIS II is on the software. Therefore the type and number of problems one can detect are related in the main to software usability and *not* to the quality of work or to the user's behaviour. EVADIS II supports expert judgement, so the workload imposed by the evaluation can be restricted. There is also computer support available, which reduces routine work. But a lot of information about the tasks and the user characteristics is needed. If it is not specified during the analysis process, the evaluation could be

very time-consuming. An expert with a grounding in human factors is needed. Whilst variations in assessment between different evaluators are reduced by a detailed evaluation guide, which describes the whole evaluation process, the final report can be biased, to a certain degree, by the judgement of the expert with respect to the relevance and rating of the evaluation items. No experimental tests are available which demonstrate the validity and the reliability of EVADIS II, and we have not made any empirical conclusions with other evaluation methods.

References

BAITSCH, C. H., KATZ, C. H., SPINAS, P. H. and ULICH, E. 1989, *Computerunterstützte Büroarbeit* (Verlag der Fachvereine, Zürich).

BONITZ, D., NACHREINER, F., BENZ, C. and WÄGER, M. 1988, Zur Veränderung von Tätigkeitsstrukturen und Arbeitsinhalten durch den Einsatz von Rechnern, *Zeitschrift für Arbeitswissenschaften*, **4**, 211–221.

BONITZ, D. 1989, *Verfahren zur Beschreibung und Bewertung von Arbeitstätigkeiten* (VBBA), Entwurf (Gesamthochschule, Kassel).

CARD, S., MORAN, T. and NEWELL, A. 1983, *The Psychology of Human–Computer Interaction*. (Lawrence Erlbaum, London).

CAKIR, A. 1991, Software-Ergonomie und Arbeitsorganisation—neue Regelungsgegenstände im Arbeitsschutz, in A. Cakir (ed.) *Europa 1992—Was bringen die Europäischen Regelwerke für Bildschirm-Arbeitsplätze?* (ERGONOMIC Institut für Arbeits- und Sozialforschung, Berlin).

CLEGG, C. W., WARR, P., GREEN, T., MONK, A., KEMP, N., ALLISON, G. and LANSDALE, M. 1988, *People and Computers—How to Evaluate Your Company's New Technology* (Ellis Horwood, Chichester).

DIN 66 234 Teil 8 1988, *Bildschirmarbeitsplätze, Grundsätze der Dialoggestaltung* (Beuth Verlag, Berlin).

DUNCKEL, H. 1989, Arbeitspsychologische Kriterien zur Beurteilung und Gestaltung von Arbeitsaufgaben im Zusammenhang mit EDV-Systemen, in S. Maaß U. H. Oberquelle (eds) *Software-Ergonomie '89—Aufgabenorientierte Systemgestaltung und Funktionalität* (Teubner Verlag, Stuttgart), 69–79.

DZIDA, W. 1983, Das IFIP-Modell für Benutzerschnittstellen, *Office-Management*, **31**, 6–8.

EASON, K. D. 1982, The process of introducing information technology, *Behaviour & Information Technology*, **1**, 197–213.

ETH-LAO 1986, *Fragebogen zur Beurteilung von Dialog-Bildschirmsystemen* (ETH Zürich, Lehrstuhl für Arbeits- und Organisationpsychologie).

EWG 1990, *Richtlinie des Rates vom 29. Mai 1990 über die Mindestvorschriften bezüglich der Sicherheit und des Gesundheitsschutzes bei der Arbeit an Bildschirmgeräten*. Fünfte Einzelrichtlinie im Sinne von Artickel 16 Absatz 1 der Richtlinie 89/391/EWG.

FÄHNRICH, K.-P. (ed.) 1987, *Software-Ergonomie* (Oldenbourg Verlag, München).

FLOYD, C. 1984, A systematic look at prototyping, in R. Budde, K. Kuhlenkamp, L. Mathiassen and H. Züllighoven (eds), *Approaches to Prototyping* (Springer Verlag, Berlin).

FRESE, M. and BRODBECK, F. 1989, *Computer in Büro und Verwaltung* (Springer Verlag, Berlin).

GRANDJEAN, E. 1987, *Ergonomics in Computerized Offices* (Taylor & Francis, London).

GREUTMANN, T. 1992, *HIDE and IDEA: Tools for User-Oriented Application Development* (Verlag der Fachvereine an den schweizerischen Hochschulen und Techniken, Zürich).

HAMMOND, N., HINTON, G., BARNARD, P., MACLEAN, A., LONG, J. and WHITEFIELD, A. 1985, Evaluating the interface of a document processor: a comparison of expert judgement and user observation, in B. Shackel (ed.) *Human–Computer Interaction—INTERACT '84* (Elsevier Science, Amsterdam), 725–729.

HOYOS, C., MÜLLER-HOLZ AUF DER HEIDE, B., HACKER, S. and BARTSCH, T. 1990, *PROTOS Menschengerechte Gestaltung von Bürokommunikationssystemen: Entwicklung von Methoden zur Herstellung und Bewertung von Prototypen für Benutzeroberflächen*, Zwischenbericht 8/90, TU München, Inst.f.Psychologie u. Erziehungswissenschaften, Lehrstuhl für Psychologie.

ISO 9241 Ergonomic Requirements for Office Work with Visual Display Terminals, Part 1, *General Introduction*, International Standard, November 1991.

ISO 9241 Ergonomic Requirements for Office Work with Visual Display Terminals, Part 2, *Guidance on Task Requirements*, International Standard, November 1989.

ISO 9241 Ergonomic Requirements for Office Work with Visual Display Terminals, Part 10, *Dialogue Principles*, Committee Draft, September 1991.

ISO 9241 Ergonomic Requirements for Office Work with Visual Display Terminals, Part 11, *Usability (Principles)*, Committee Draft, January 1992.

KISHI, N. and KINOE, Y. 1991, Assessing usability evaluation methods in a software development process, in H.-J. Bullinger (ed.) *Human Aspects in Computing: Design and Use of Interactive Systems and Work with Terminals* (Elsevier, Amsterdam), 597–601.

KOCH, M., REITERER, H. and TJOA, A. 1991, *Software-Ergonomie, Gestaltung von EDV-Systemen-Kriterien, Methoden und Werkzeuge* (Springer Verlag, Wien).

KÖCHLING, A. 1990, *Gestaltungswerkzeug Checkliste Bildschirmergonomie* (Forkel Verlag, Wiesbaden).

LANG, J. and PETERS, H. 1988, *Erhebung ergonomischer Anforderungen an Software, die überprüfbar und arbeitswissenschaftlich abgesichert sind* (Institut für Software-Ergonomie, TÜV Bayern).

MAMBREY, P., OPPERMANN, R. and TEPPER, A. 1986, *Computer und Partizipation. Ergebnisse zu Gestaltungs- und Handlungspotentialen* (Westdeutscher Verlag, Opladen).

MITRE 1991, *Dynamic Rules for User Interface Design (DRUID, Version 2.0)* (MITRE Corporation, Bedford).

MITRE 1986, *Guidelines for Designing User Interface Software*, (MITRE Corporation, Bedford).

MUSiC 1992, *Metrics for Usability Standards in Computing* (ESPRIT II Project 5429), Product information (National Physical Laboratory, UK).

NORMAN, K. and SHNEIDERMAN, B. 1989, *Questionnaire for*

User Interaction Satisfaction (QUIS 5.0) (HCI Lab, College Park, University of Maryland).

OLLE, T., HAGELSTEIN, J., MACDONALD, I., ROLLAND, C., SOL, H. and ASSCHE, F. V. 1988, *Information Systems Methodologies* (Addison-Wesley, Wokingham).

OPPERMANN, R., MURCHNER, B., REITERER, H. and KOCH, M. 1992, *Software-ergonomische Evaluation, Der Leitfaden EVADIS II* (Walter de Gruyter Verlag, Berlin).

OPPERMANN, R., MURCHNER, B., PAETAU, M., PIEPER, M., SIMM, H. and STELLMACHER, I. 1989, *Evaluation of Dialog Systems*, (GMD-Studie Nr. 169, St Augustin).

PIEPENBURG, U. and RÖDIGER, K.-H. 1989, *Mindestanforderungen an die Prüfung von Software auf Konformität nach DIN 66234, Teil 8*, Werkstattbericht Nr. 61 der Reihe 'Mensch und Technik—Sozialverträgliche Technikgestaltung', Ministerium für Arbeit, Gesundheit und Soziales.

RAVDEN, S. and JOHNSON, G. 1989, *Evaluating Usability of Human–Computer Interfaces: A Practical Method* (John Wiley, New York).

REITERER, H. 1990, Ergonomische Kriterien für die menschengerechte Gestaltung von Bürosystemen, Anwendung und Bewertung, Dissertation, Universität Wien.

ROBERTS, T. and MORAN, T. 1983, The evaluation of text editors: methodology and empirical results, *Communications of the ACM*, **26**, 265–283.

RÖDIGER, K.-H., NULLMEIER, E. and OESTERREICH, R. 1986, *Verfahren zur Ermittlung von Regulationserfordernisse in der Arbeitstätigkeit im Büro (VERA/B)* (Berlin).

RUDOLPH, E., SCHÖNFELDER, E. and HACKER, W. 1987, *Tätigkeitsbewertungssystem—Geistige Arbeit (TBS-GA)* (Hogrefe Verlag, Göttingen).

SHERWOOD-SMITH, M. 1989, The evaluation of computer-based office systems, Dissertation, National University of Ireland.

SIEMENS 1987, *Handbuch für Prüftechnik, Prüfliste Benutzerfreundlichkeit* (Siemens, München)

SIEMENS 1990, *Styleguide-Checkliste* (Siemens Nixdorf Training Center, München).

SIMES, D. and SIRSKY, P. 1985, Human factors: an exploration of the psychology of human–computer dialogues, in R. Hartson (ed.) *Advances in Human–Computer Interaction*, Volume 1 (Ablex, Norwood), 49–103.

SPINAS, PH. 1987, *Arbeitspsychologische Aspekte der Benutzerfreundlichkeit von Bildschirmsystemen*, (ADAG Administration & Druck, Zürich).

TBS 1991, *Der Software Prüfer—Ein Leitfaden zur Bewertung von Dialogprogrammen* (Technologieberatungsstelle Hannover).

TRIEBE, J. K., WITTSTOCK, M. and SCHIELE, F. 1987, *Arbeitswissenschaftliche Grundlagen der Software-Ergonomie* (Schriftreihe der Bundesanstalt für Arbeitsschutz, S 24, Dortmund).

TULLIS, TH. 1988, A system for evaluating screen formats: research and application, in R. Hartson and H. Deborah (eds) *Advances in Human–Computer Interaction*, Volume 2 (Ablex, Norwood), 214–286.

WHITEFIELD, A., WILSON, F. and DOWELL, J. 1991, A framework for human factors evaluation, *Behaviour & Information Technology*, **1**, 65–79.

ZÖLCH, M. and DUNCKEL, H. 1991, Erste Ergebnisse des Einsatzes der 'Kontrastiven Aufgabenanalyse', in D. Ackermann u. E. Ulich (eds) *Software-Ergonomie '91, Benutzerorientierte Software-Entwicklung* (Teubner Verlag, Stuttgart), 363–372.