in: Design, Specification and Verification of Interactive Systems '95. (P. Palanque & R. Bastide, eds.) Wien New York: Springer, 1995, pp. 209-223.

Four different measures to quantify three usability attributes: 'feedback', 'interactive directness' and 'flexibility'

Matthias Rauterberg

Work and Organisational Psychology Unit (IfAP)
Swiss Federal Institute of Technology (ETH)
Nelkenstrasse 11, CH–8092 Zuerich, Switzerland
Tel: +41-1-63-27082, Email: rauterberg@ifap.bepr.ethz.ch

Abstract. One of the main problems of standards (e.g., DIN 66234, ISO 9241) in the context of usability of software quality is, that they can not be measured in product features. We present a new approach to measure user-interface quality in a quantitative way. First, we developed a concept to describe user-interfaces on a granularity level, that is detailed enough to preserve important interface characteristics, and is general enough to cover most of known interface types. We distinguish between different types of 'interaction-points'. With these kinds of interaction-points we can describe several types of interfaces (CUI: command, menu, form-fill-in; GUI: desktop, direct manipulation, multimedia, etc.). We carried out two different comparative usability studies to validate our quantitative measures. The results of one other published comparative usability study can be predicted. Results of six different interfaces are presented and discussed.

Keywords: user-interfaces, utility functions, testability, quantification

1 Introduction

One of the main problems of standards (e.g., DIN 66234, ISO 9241 Part 10) to quantify software quality of usability is, that they can not be measured in product features [11]. Four different views on human computer interaction to measure interactive qualities currently exists (cf. [1], [17]).

(1) The *interaction-oriented view:* usability quality is measured in terms of how the user interacts with the product ("usability testing"). This view is the most common one. All kinds of usability testing with "real" users are subsumed in this category [8].
(2) The *user-oriented view:* usability quality is measured in terms of the mental effort and attitude of the user ("questionnaires" and "interviews").
(3) The *product-oriented view:* usability quality is measured in terms of the ergonomic attributes of the product itself (quantitative measures).
(4) The *formal view:* usability is formalised and simulated in terms of mental models (formal concepts). Karat [10] describes formal methods in the context of "theory-based" evaluation.

The interactive qualities of user-interfaces currently are quantified in the context of *interaction-oriented view* and *user-oriented view*, but these both approaches are time consuming and more or less expensive [9].

It would be helpful if usability attributes could be quantified in such a way that the extent of each attribute could be measured in product features. Levels of measurement

are classified in different scales as follows (cf. Tab. 1): (1) *nominal scale* (to classify or grouping interfaces), (2) *ordinal scale* (to compare different types of interfaces and to put categories in order), (3) *interval scale* (meaningful measure of the distance between categories), and (4) *rational scale* (interval scale with an absolute null) (cf. [12]).

Tab. 1. Levels of measurement of usability attributes

level	*examples in the context of Human-Computer Interaction*	*reference*
nominal	Type of interface (e.g., command, menu, desktop, etc.)	[18]
ordinal	Experimental comparison study (e.g., summative evaluation)	[3] [7] [15]
interval	Checklist (e.g., expert evaluation)	[11] [18]
rational	Quantitative measure	[16]

2 A descriptive concept of interaction-points

We present a new approach to measure user-interface quality in a quantitative way. First, we developed a concept to describe user-interfaces on a granularity level, that is detailed enough to preserve important interface characteristics, and is general enough to cover most of known interface types (command language, CUI, GUI, multimedia, etc.). Different types of user-interfaces can be quantified and distinguished by the general concept of "interaction-points". Regarding to the interactive semantic of "interaction-points" (IPs), different types of IPs must be discriminated (cf. [4]).

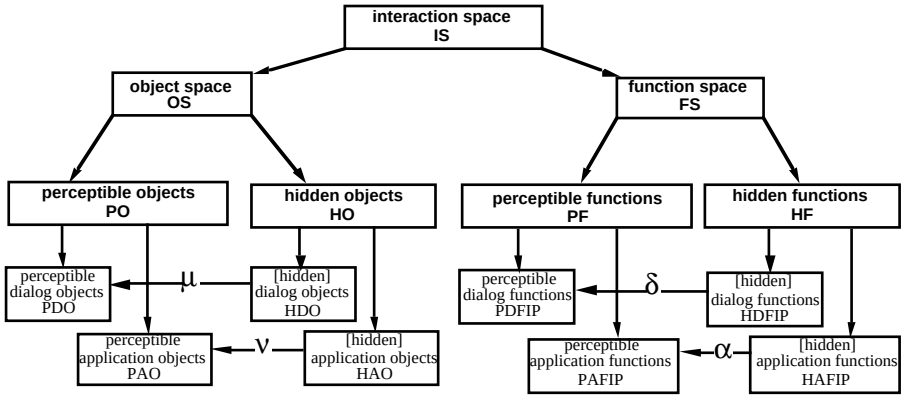

Fig. 1. The interaction space (IS) consists of the object space (OS) and the function space (FS); both spaces can be distinguished in perceptible and hidden interactive elements.

An interactive system can be distinguished in a dialog and an application manager. So, we distinguish between dialog objects (DO, e.g. "window") and application objects (AO, e.g. "text document"), and dialog functions (DF, e.g. "open window") and application functions (AFIP, e.g. "insert section mark"). Each function $f \in FS$, that changes the state of the content of an application object, is an *application function*. All other functions are *dialog functions* (e.g., window operations like move, resize,

close). The complete set of all description terms is shown in Fig. 1 and defined in Tab. 2.

Tab. 2. The interaction space (IS) consists of the object (OS) and the function (FS) space

IS := OS x FS	[interaction space]
DC ∈ IS	[dialog context]
OS := PO ∪ HO	[object space]
FS := PF ∪ HF	[function space]
PO := PDO ∪ PAO	[(perceptible) representations of objects]
HO := HDO ∪ HAO	[hidden objects]
PF := PDFIP ∪ PAFIP	[(perceptible) representations of functions]
HF := HDFIP ∪ HAFIP	[hidden functions]
PDFIP := {(df, pf) ∈ HDFIP x PF: pf = δ(df)}	[(perceptible) represented DFIP]
PAFIP := {(af, pf) ∈ HAFIP x PF: pf = α(af)}	[(perceptible) represented AFIP]
IP := DFIP ∪ AFIP	[interaction-points]
DFIP := PDFIP ∪ HDFIP	[IPs of dialog functions]
AFIP := PAFIP ∪ HAFIP	[IPs of application functions]
δ := mapping function of a df ∈ HDFIP to an appropriate pf ∈ PF.	
α := mapping function of an af ∈ HAFIP to an appropriate pf ∈ PF.	
PDO := {(do, po) ∈ HDO x PO: po = μ(do)}	[(perceptible) represented DO]
PAO := {(ao, po) ∈ HAO x PO: po = ν(ao)}	[(perceptible) represented AO]
μ := mapping function of a dialog object do ∈ DO to an appropriate po ∈ PO.	
ν := mapping function of an application object ao ∈ AO to an appropriate po ∈ PO.	

A dialog context (DC) is defined by all available objects and functions in the actual system state. If the set of available functions changes in the actual DC, then the system changes from one DC to another. In the actual DC all dialog objects (functions, resp.) are *perceptible* (PO, PF) or *hidden* (HO, HF). Four different mapping functions relate perceptible structures to hidden objects or functions (see Fig. 1).

Each interaction-point (IP) is related to at least one interactive function. If both mapping function's δ and α are of the type 1:m(any), then the user-interface is a command interface (see Fig. 2). If both mapping function's δ and α are of the type 1:1, then the user-interface is a menu or direct manipulative interface where each f∈FS is related to a perceptible structure PF (see Fig. 3). The perceptual structure (visible, audible, or tactile) of a function (PF) can be, e.g., an icon, earcon, menu option, command prompt, or other mouse sensitive areas.

The intersection of PF and PO is sometimes not empty: PF ∩ PO ≠ ∅. Icons of graphical interfaces are elements of this intersection, e.g., PDFIP "copy" ≡ PDO "clipboard", PAFIP "delete" ≡ PAO "trash" (see Fig. 3). Each interaction-point (IP) is related to at least one interactive function (see Fig. 4).

3 Four quantitative measures of interface attributes

One important difference between user-interfaces can be the "interactive directness". A user-interface is 100% *interactively direct*, if the user has fully access in the actual dialog context to all f∈FS (see [13], e.g. [20]). This is the case for all command language interfaces (cf. Fig. 2). Another important interface attribute is the amount of "feedback". Good interface design is characterised by optimising the multitude of

DFIPs (e.g. "flatten" the menu tree; see [14]) and by allocating an appropriate PDFIP to the remaining HDFIPs (cf. [2]).

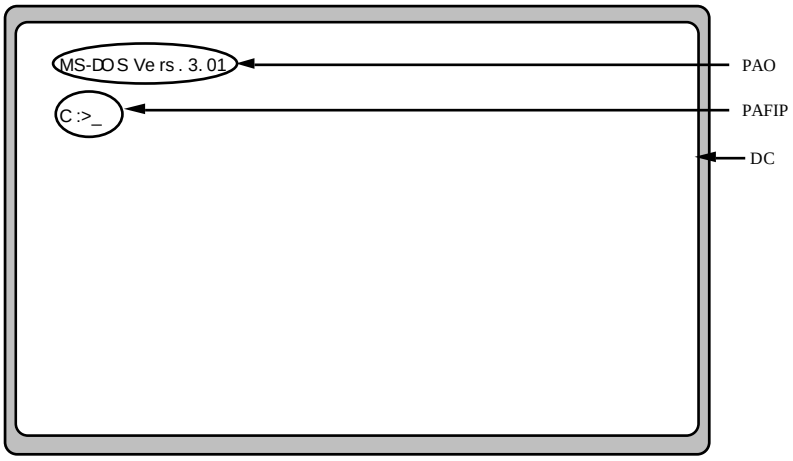

Fig. 2. An actual dialog context (DC) of the operating system MsDOS with a command language interface (PAFIP: command entry point).

One disadvantage of snapshots (cf. Fig. 2 and Fig. 3) is that all hidden structures could not be referenced. To describe the hidden functionality a schematic view is needed (cf. Fig. 4).

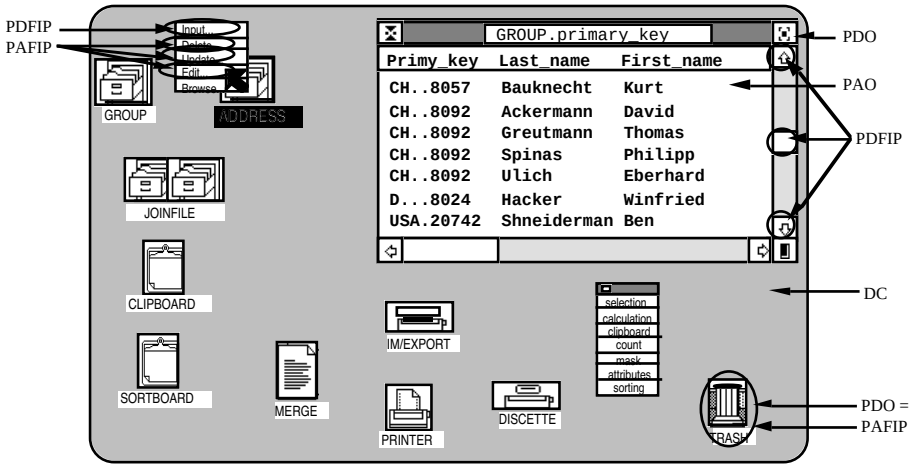

Fig. 3. An actual dialog context (DC) of a direct manipulative interface with the representation space of the interactive object (PAO: e.g., data window; PDO: e.g., trash), and the representation space (PF: marked by circles) of the interactive functions (PAFIP: e.g., pop-up menu, trash; PDFIP: e.g., window scrolling).

To estimate the amount of "feedback" of an interface a ratio is calculated: "number of PFs" (#PF = #PDFIP + #PAFIP) divided by the "number of HFs" (#HF = #HDFIP +

#HAFIP) per dialog context. This ratio quantifies the average "amount of functional feedback" of the function space (FB; see Formula 1). We abbreviate the number of all different dialog contexts with D. A GUI has often a very large number of DCs. To handle this problem we take only all task related DCs into account. Doing this, our measures will give us only a lower estimation for GUIs.

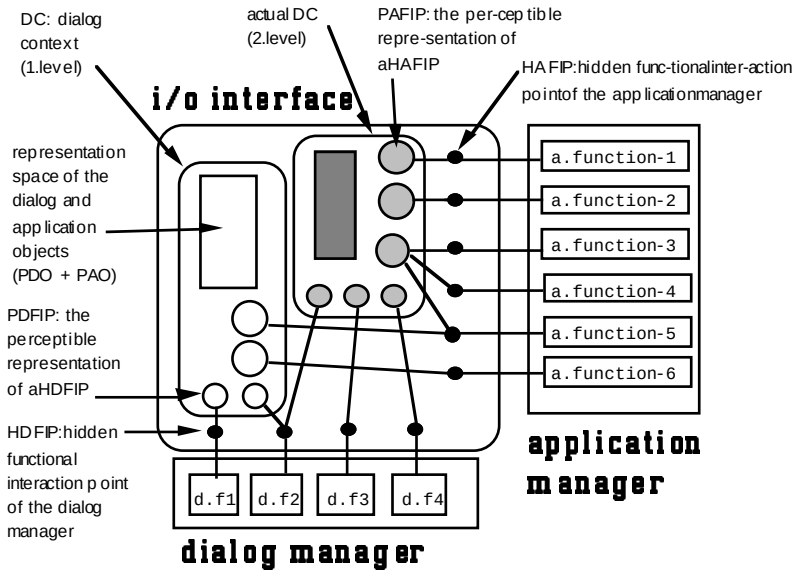

Fig. 4. A schematic presentation of the I/O interface, the dialog and the application manager of an interactive system with a menu tree of two levels.

The average length of all possible sequences of interactive operations (PATH) from the top level dialog context (DC, e.g., 'start context') down to DCs with the desired HAFIP or HDFIP can be used as a possible quantitative measure of "interactive directness" (ID, see Formula 2). The measure ID delivers two indices: one for HAFIPs and one for HDFIPs. A PATH has no cycles and has not more than two additional dialog operations compared with the shortest sequence. An interface with the maximum ID of 100% has only one DC with path lengths of one dialog step. We abbreviate the number of all different dialog paths with P.

Functional feedback:
$$FB = \frac{1}{D}\sum_{d=1}^{D}\left(\#\mathrm{PF}_d / \#\mathrm{HF}_d\right) * 100\% \qquad (1)$$

Interactive directness:
$$ID = \left\{\frac{1}{P}\sum_{p=1}^{P}\lg\left(\mathrm{PATH}\,p\right)\right\}^{-1} * 100\% \qquad (2)$$

To quantify the flexibility of the application manager we calculate the average number of HAFIPs per dialog context (DFA; see Formula 3). To quantify the flexibility of the dialog manager we calculate the average number of HDFIPs per dialog context

(DFD; see Formula 4). A modeless dialog state has maximal flexibility (e.g., "command" interfaces, or Oberon; [20]).

Application flexibility: $$DFA = \frac{1}{D}\sum_{d=1}^{D}(\#\,HAFIP_d) \quad (3)$$

Dialog flexibility: $$DFD = \frac{1}{D}\sum_{d=1}^{D}(\#\,HDFIP_d) \quad (4)$$

Let us apply the five measures to our example in Fig. 4. The average amount of functional feedback is:
FB = (4/4 + 6/9) / 2 * 100% = 83.3 %.

The average amount of interactive directness is:
ID_{HAFIP} = ((2*1 + 5*2) / 7)$^{-1}$ * 100% = 58.3 %;
ID_{HDFIP} = ((2*1 + 3*2) / 5)$^{-1}$ * 100% = 62.5 %.

The average amount of flexibility is:
DFA = (2 + 5) / 2 = 3.5 and
DFD = (2 + 3) / 2 = 2.5.

To interpret the results of our measure's appropriately empirical studies are necessary.

4 Results and discussions of three empirical studies

We carried out two different comparative usability studies to validate our measures ([3] [15]). A third external comparative study [7] was used for a cross-validation. All three investigated software products have the same application manager, but two different dialog managers each.

4.1 Experiment-I

Method. Rauterberg [15] compared a traditional menu-driven interface (character-oriented user-interface: CUI; cf. Fig. 5) of a relational database management system with a modern desktop interface (graphic-oriented user-interface: GUI; cf. Fig. 6) of the same application manager. We chose this program, because there are two different types of interfaces for the same database machine running on the same hardware (DOS PCs). Both types of interfaces are distributed as standard software on the European market.

Subjects. Twelve paid beginners (novice and naive, see [5]) took part in this study. The twelve experts (experienced and expert) had been working with the respective user-interface for several years in their daily work; they were chosen from the address list of the software company. The experts did not receive any payment; the beginners were paid. The previous experience was carefully measured with a questionnaire during a semi structured interview (average duration for the interviews with the experts: 40-50 min).

CUI-beginners: average age of 27 years; 4 women, 2 men; 31 hrs. of general previous experience with EDP; 1.5 hrs. of instruction.

CUI-experts: average age of 38 years; 6 men; 7.500 hrs. of general previous experience with EDP; 1.736 hrs. of experience with specific user-interface (menu-selection).

GUI-beginners: average age of 21years; 2 women, 4 men; 68 hrs. of general previous experience with EDP; 1.5 hrs. of instruction.

GUI-experts: average age of 38 years; 6 men; 3.700 hrs. of general previous experience with EDP; 1.496 hrs. of experience with specific user-interface (desktop).

Independent Factors. The test design is characterised by three independent factors: (1) two diffent interfaces (Factor-A: CUI vs. GUI), (2) ten different tasks (Factor-B: task-1, task-2, …, task-10), and (3) the different experience of the test subjects (Factor-C: beginners vs. experts).

The CUI interface consists of a strict hierarchical menu tree with exact three levels (like MsWord, cf. Fig. 3). Starting with the main menu (level 1) the user can make active each module (level 2) by pressing the corresponding letter key. In the dialogue context of a module the different routines (level 3) can be activated by pressing another letter key. On main menu or module level only, help, global switches, the active data base file, and redirection input or output could be activated or changed using the function keys. In an activated routine context the dialogue control could be achieved alone by pressing function keys.

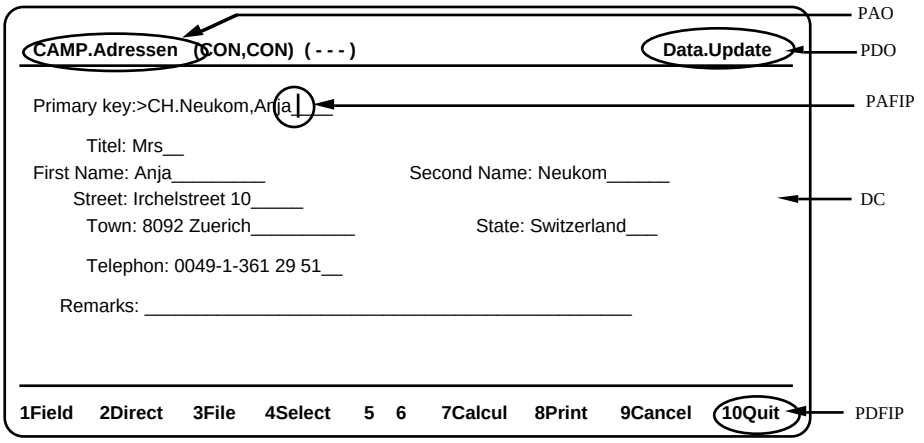

Fig. 5. The CUI interface of the data base management system with the representation space of the interactive objects (PAO: e.g., data file name; PDO: e.g., module name.routine name), and the representation space of the interactive functions (PAFIP: e.g., text entry point; PDFIP: e.g., function key F10).

The GUI interface is implemented under GEM on DOS PCs (cf. Fig. 6). The screen of the desktop interface is divided into four different areas. The row at the top of the screen contains the semantic labels of all pull-down menus (first area). The second row just below is an output or 'info area' (second area). The biggest part is the desktop area with all icons, windows, and dialogboxes (third area). The bottom row is a string of all semantic labels for all function keys. Each label field (F1 to F10) is a mouse sensitive area, too (fourth area). The dialogue of this typical type of desktop interface can be controlled by pull-down menus and partly by function keys, as well.

Dependent measures. The dependent variable was the pure 'task solving time' of each user according to logfile record excluding system response time; the control variables were: the number of hours of general previous experience with EDP (included specific experience with the two different systems).

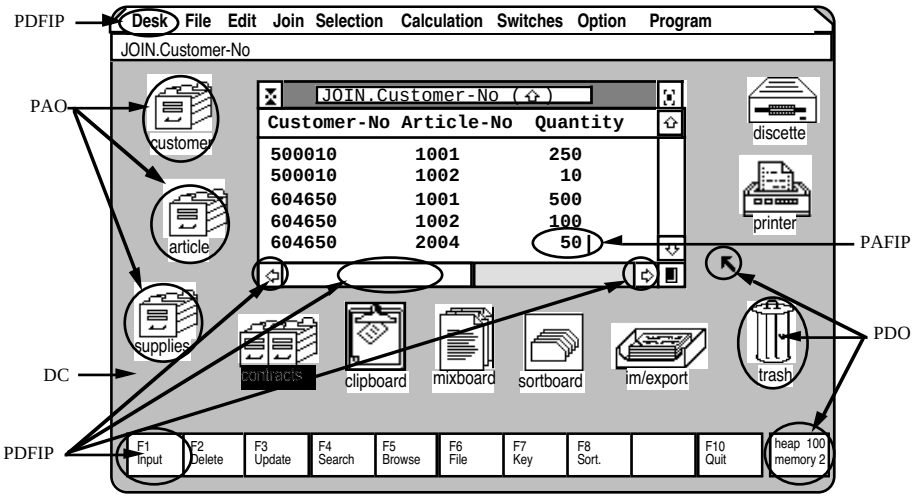

Fig. 6. The GUI interface of the data base management system with the representation space of the interactive objects (PAO: e.g., data files; PDO: e.g., mouse pointer, trash), and the representation space of the interactive functions (PAFIP: e.g., data entry point; PDFIP: e.g., pull down menu, function key F1, scroll bar areas).

Tasks. Ten tasks were selected according to whether they allowed the subjects to use exactly the same functionality of the application manager with both types of interfaces and to use those operations which are most common in daily data base work. The test data base consisted of three files, which contained all necessary attributes to manage a camping place. Tasks 9 and 10 were selected to test whether the design of the user-interface was appropriate for these types of tasks, too.

Task 1: "Please find out how many data records are in the file ADDRESS, in the file PLACE, and in the file GROUP." The user has to activate a specific menu option ("File info..." of the desktop interface; resp. "Datafile" in module "Info" of the menu interface) and to read the file size (file: ADDRESS - 280 data records, PLACE - 17 data records, GROUP - 27 data records).

Task 2: "Delete only the last data record of the file ADDRESS, the file PLACE, and the file GROUP (sorted by the attribute 'namekey')." The user has to open (sorted according to the given attribute), select and delete the last data record (file: PLACE, ADDRESS, GROUP).

Task 3: "Search and select the data record with the namekey 'D..8000C O M' in the file ADDRESS, and show the content of all attributes of this data record on the screen. Correct this data record for the following attributes: State: D, Place offer: 07, Remarks: The system dealer can give a demonstration." The user must select a certain data record (file: ADDRESS), correcting the data record with regard to three attributes.

The seven remaining tasks are completely described in [15].

Results. At the start the results were analysed from the point of view of a three-factorial design across only the first six tasks (Factor-A "interface", Factor-B "task (1-6)", and Factor-C "expertness"). In the 90 minutes most of the beginners were not able to finish more than the first six (or eight at most) of the tasks given, so that the tasks 8 to 10 were excluded from this analysis. Almost all experts were able to finish all the ten tasks given.

Tab. 3. Results of the dependent variable 'task solving time' of the comparison study CUI vs. GUI; the alpha-error is abbreviated with p

Dependent variable: *'Task solving time'*	Mean ± Standard Deviation *CUI (N=6)*	*GUI (N=6)*	Effect *p*
beginners	1073 s ± 590 s	670 s ± 490 s	< .002
experts	414 s ± 245 s	201 s ± 137 s	< .001
total	683 s ± 556 s	418 s ± 437 s	< .001

The main result of this empirical investigation was, that the mean task solving time with the GUI (experts: 201 s, beginners: 670 s) is significantly shorter than with the CUI (experts: 414 s, beginners: 1073 s) interface for beginners and experts, too (see Tab. 3). For all the first six tasks the users of the GUI interface needed less time to solve the tasks than the users with the CUI interface.

Discussion. Contrary to the often voiced opinion that a desktop interface with direct manipulation is good for beginners only, it is the expert group with the desktop interface who has profit of GUI's, too. Compared to them the beginners with menu selection had particularly bad results.

No specific dependency between the previous experience with EDP and the different types of tasks can be assumed. Generally it was not clear in which way the restriction on database handling applied in this investigation was responsible for the results found in this study.

On the whole it is to be emphasised that a desktop interface with direct manipulation by means of the "mouse" as a general element of interaction was superior to the conventional user-interface with menu selection by means of the "function keys". This is true particularly for users with long previous database experience (experts).

How can we explain the observed advantage of the GUI? Our first interpretation of this outcome was the supposed different amount of 'transparency' [19]. One aspect of 'transparency' is 'feedback' (see [6] pp. 318-321). But, if we take the results of our quantification into account, then we can assume that the different amount of flexibility is the critical quality.

4.2 Experiment-II

Method. The second experiment was run on a PC with colour screen. The standard Windows 3.0 environment with a multimedia information system of a German bank association with a hierarchical dialog structure was used. The original version was developed by the German multimedia software house ADI Inc. in Karlsruhe (D). The second version of this multimedia system was redesigned and programmed at our usability laboratory to get a system with a net-shaped dialog structure.

Subjects. A total of 12 beginners participated. Group-A consists of one woman and five men with the average age 24.2 ± 0.4 years. Group-B consists of two women and

four men with the average age 22.5 ± 0.8 years. Group-A tested first the original version and in a second trial the redesigned version. Group-B tested both systems in reverse order.

Independent Factors. The test design is characterised by three independent factors: (1) two different interfaces (Factor-A: graphical interface with a hierarchical dialog structure GUI_{hier} vs. graphical interface with a net-shaped dialog structure GUI_{net}), (2) nine different tasks (Factor-B: task-1, task-2, …, task-9), and (3) the different sequence of testing both systems (Factor-C: GUI_{hier} –>GUI_{net} vs. GUI_{net} –>GUI_{hier}).

The original version consists of 62 different screens (masks) with on average 11.6 ± 5.1 objects per screen (number of all objects is 721; cf. Fig. 7). The second version consists of 51 different screens (masks) with on average 13.2 ± 4.9 objects per screen (number of all objects is 672).

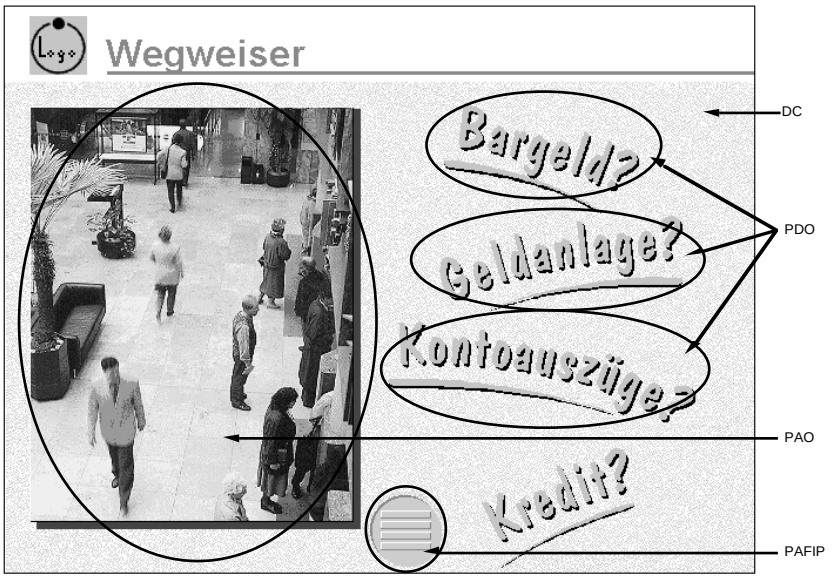

Fig. 7. The graphical interface of the multimedia information system with the representation space of the interactive objects (PAO: e.g., picture of entrance hall; PDO: e.g., several topics), and the representation space of the interactive functions (PAFIP: e.g., button to next screen).

Dependent measures. The two dependent variables were 'task solving time' per task and 'number of masks successions' over all nine tasks.

Tasks. Subjects were instructed to solve nine tasks: «(1) Search a house for a price of 450,000.– DM. (2) Who is responsibly for the sales talk about an estate? (3) Where is the office of this person located in the building? (4) To buy the house you need a mortgage. Where can you get this? (5) Where can you get information about buying and selling of securities? (6) The bank offers different events of entertainment. You have a free day (April, 7th, 1993). Which events are offered? (7) You have not enough cash and you are nearby the main station. Where is the next cash service? (8) Where is the cash counter located in the building? (9) Which spectrum of services are available at the cash service desk?»

Results. The multimedia information system with the net-shaped dialog structure (GUI_{net}) is not superior to the system with the hierarchical dialog structure (GUI_{hier}). It seems to be that the users of the 'more flexible' system need more time to solve the tasks (cf. Tab. 4).

To make sure that this result is not biased by an unknown aspect the experiment was carried out a second time. The results are exactly the same as in the first investigation.

Tab. 4. Results of the two dependent variables of the comparison study GUI_{hier} vs. GUI_{net}; the alpha-error is abbreviated with p

	Mean ± Standard Deviation		Effect
Dependent variable:	*GUI_{hier} (N=6)*	*GUI_{net} (N=6)*	*p*
'Task solving time'	9.7 min ± 3.8 min	10.8 min ± 4.3 min	< .085
'#masks successions'	54 ± 15 masks	56 ± 19 masks	< .625

Discussion. The comparison of both multimedia interfaces could not show an empirical provable difference between the hierarchical and the net-shaped dialog structure. The amount of feedback is for both interfaces identically: each HF has at least one PF. If we could show a performance difference then this difference must be caused by the type of dialog structure. But, we can not find a difference between both versions in task solving time or in number of masks successions (cf. Tab. 4).

4.3 Experiment-III

Method. The study of Grützmacher [7] was carried out to investigate research questions in the context of how to control a complex domain ('development aid for a fictive society in Africa') with the simulation tool "Moro". Two different dialog structures (hierarchical vs. net-shaped) with the same CUI interface were compared (cf. Fig. 8). The program was implemented on a host computer and could be used during six months. This host computer was installed for public access at the University of Zurich. Each session was automatically recorded. Only the first simulation session is included in the further analysis. All second or more sessions are excluded.

Subjects. The sample consists of 20 unknown users with the hierarchical dialog structure (average age 25 ± 3 years) and 15 unknown users with the net-shaped structure (average age 28 ± 6 years).

Independent Factors. One independent factor was varied: the dialog structure (Factor A: hierarchical CUI_{hier} versus net-shaped CUI_{net} dialog structure). This simulation tool was implemented on a mainframe computer system with character oriented terminals (IBM 3270). The simulation tool was controlled by several parameters (e.g., 'number of inhabitants', 'number of cows', 'capital stock', etc.).

Dependent measures. At the end of each simulation run the absolute difference between the actual value and the targeted value of all eight parameters was calculated and divided by the targeted value ('percental difference'). The dependent variable 'target discrepancy' was the average of all eight percental differences.

Tasks. The user was instructed to meet several targets, one for each of eight different parameters (e.g., 'number of inhabitants' = 800, 'number of cows' = 4000, etc.). To control the simulation the user could change 49 different parameters. In CUI_{net} each of all 49 parameters was presented in exactly one mask per parameter. In CUI_{hier} several parameters could be changed in more than one mask.

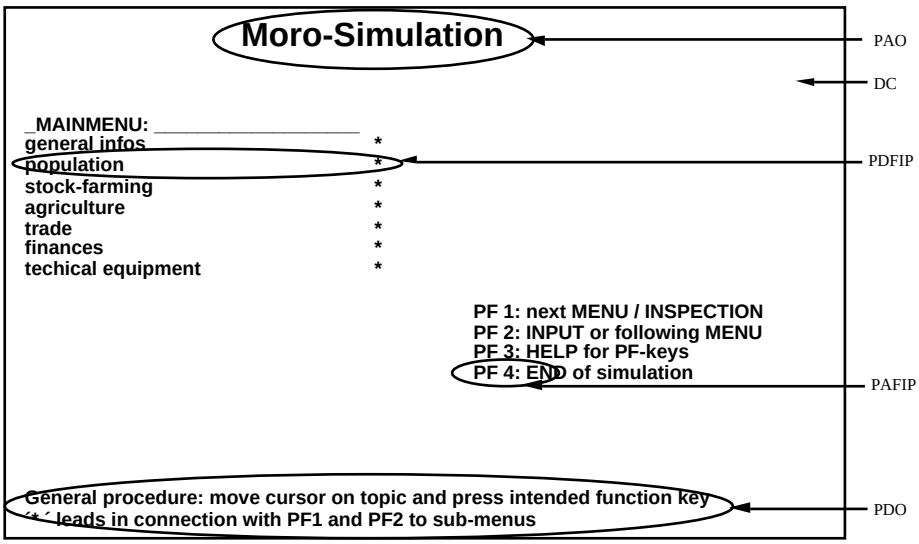

Fig. 8. The CUI-interface of the simulation tool with the representation space of the interactive objects (PAO: e.g., name of the tool; PDO: e.g., context sensitive help text), and the representation space of the interactive functions (PAFIP: e.g., function key END; PDFIP: e.g., menu option).

Results. The net-shaped dialog structure was not superior to the hierarchical dialog structure (cf. Tab. 5). This study showed similar results as in the second experiment.

Tab. 5. Results of the dependent variable 'target discrepancy' of the comparison study CUI_{hier} vs. CUI_{net}; the alpha-error is abbreviated with p

	Mean ± Standard Deviation		Effect
Dependent variable:	*CUI_{hier} (N=20)*	*CUI_{net} (N=15)*	*p*
'Target discrepancy'	49% ± 15%	48% ± 18%	< .825

The average number of 'simulation years' was for CUI_{hier} = 19 ± 7 and for CUI_{net} = 27 ± 10 'years' (T-test, $p < .043$). The users played significantly 'longer' with CUI_{net} than with CUI_{hier}.

Discussion. The hypothesis of Grützmacher [5] was that the net-shaped dialog structure is superior to the hierarchical structure. The study showed no differences between both dialog structures measured with "target discrepancy" as a performance measure. Grützmacher presented no explanation for this negative result.

5 Results of applying the measures

Interesting is the fact, that the GUI of experiment-I supports the user with less "visual feedback" (FB = 66%, see Tab. 6) on average than the CUI (FB = 73%). This amount of FB of the CUI is caused by 22 small DCs with FB = 100%; the GUI has only 14 DCs with FB = 100%. The amount of functional feedback seems not to be related to the advantage of GUIs. There must be another reason.

The "interactive directness" is not quite different between both interfaces:
CUI: ID = 24.7% for AFIPs and 23.2% for DFIPs versus
GUI: ID = 22.5% for AFIPs and 25.5% for DFIPs (see Tab. 6).

Only the two measures of "flexibility" show an important difference:
CUI: DFA = 12.1 and DFD = 10.1 versus
GUI: DFA = 19.5 and DFD = 20.4 (see Tab. 6).

In the hierarchical dialog structure of the multimedia information system (experiment-II) only one way is given to reach an AFIP. In the net-shaped version several ways are possible to navigate through the dialog structure. But, what is an AFIP in the context of a multimedia system? What is the application kernel?

We define the *application kernel* of a multimedia information system as the set of all masks with a relevant information in the sense of the main purpose of the information system (e.g., concrete information's about bank services in the context of a bank information system); all other masks are part of the dialog manager. A PAFIP is therefore each mouse sensitive area that changes the system to a mask of the application kernel; all other buttons or mouse sensitive areas are DFIP's.

Tab. 6. Comparison our three empirical validation studies relating to the quantitative measures ID, FB, DFA, and DFD. P is the number of all different dialog PATHs for an AFIP or a DFIP; D is the number of all different DCs

Experiment	Interface type and dialog structure	$P_{(AFIP)}$	$ID_{(AFIP)}$ %	$P_{(DFIP)}$	$ID_{(DFIP)}$ %	D	FB %	DFA	DFD
I	CUI-hierarchical	434	24.7	362	23.2	36	73	12.1	10.1
I	GUI-hierarchical	547	22.5	570	25.5	28	66	19.5	20.4
II	Multimedia-hierarchical	241	25.1	34	28.1	68	100	3.6	0.5
II	Multimedia-net shaped	276	40.7	87	46.3	65	100	4.2	1.3
III	CUI-hierarchical	720	20.9	693	23.9	363	86	2.0	1.9
III	CUI-net shaped	490	15.8	1053	21.9	389	90	1.3	2.7

With the generous support of Grützmacher we were able to analyse all 752 dialog contexts for both interfaces of the simulation tool 'Moro' (cf. experiment-III). For the hierarchical CUI we get the following results: DFA = 2.0 and DFD = 1.9; for the net-shaped CUI: DFA = 1.3 and DFD = 2.7 (see last two rows in Tab. 6). These results for DFA and DFD of both CUI interfaces give us a strong empirical evidence that the following assumptions are correct:

(1) The dialog flexibility can be quantitatively measured in a task independent way, and
(2) the values of DFA and DFD must exceed the threshold of 15.

6 Discussion

If our interpretation of the outcome of experiment-I is correct then we can not find a significant performance difference for dialog structures that remain under the assumed threshold of 15. To control the factor of feedback we carried out the second experiment with a multimedia information system that has 100% functional feedback for both interfaces [3].

We picked out a multimedia information system with a hierarchical dialog structure where DFA and DFD are clearly under 15. We implemented a comparable system with a net-shaped dialog structure where DFA and DFD have nearly the same ratio of flexibility as in experiment-I:

$DFA_{GUI} / DFA_{CUI} = 1.6$ and $DFA_{MMnet} / DFA_{MMhier} = 1.2$;

$DFD_{GUI} / DFD_{CUI} = 2.0$ and $DFD_{MMnet} / DFD_{MMhier} = 2.6$.

As we predicted, we can not find a significant performance difference between both types of dialog structures (see Tab. 4). To make sure that our results are not biased by our own expectations, we carried out a cross validation study. To do this, (1) we need the outcomes of an external independent comparison study between two different interfaces and (2) the possibility to apply our quantitative measures to all DCs of both interfaces. The empirical investigation of Grützmacher [7] fulfils both conditions.

Given our interpretation of the last two experiments we expected and found a value for DFA and DFD under 15. We interpret the negative result of experiment-III to the effect that flexibility must exceed a threshold to be effective (DFD, DFA > 15).

7 Conclusion

Using the quantitative measures for "feedback", "interactive directness" and "flexibility" to measure the interactive quality of user-interfaces, we are able to classify the most common types: command, menu, desktop. The command interface is characterised by high interactive directness, but this interface type has a very low amount of visual feedback. Especially graphical interfaces (e.g., multimedia) can support users with sufficient interactive directness. GUIs are characterised by high dialog flexibility.

The presented approach to quantify usability attributes and the interactive quality of user-interfaces is a first step in the right direction. The next step is a more detailed analysis of the relevant characteristics and validation of these characteristics in further empirical investigations. In the context of standardisation we can use our criteria to test user-interfaces for conformity with standards.

Acknowledgements: We have to thank the following persons for their generous support: Dr. Karl Schlagenhauf and Raimund Mollenhauer at ADI Software Inc., Karlsruhe (D), Prof. Kurt Bauknecht at the University of Zürich, Andreas Grützmacher and Prof. Eberhard Ulich at the ETH, Zürich (CH).

8 References

1. Bevan, N., J. Kirakowski and J. Maissel: What is Usability? In: H-J. Bullinger (ed.): Human Aspects in Computing: Design and Use of Interactive Systems with Terminals. Amsterdam: Elsevier 1991, pp. 651-655.
2. Bodart, F. and Vanderdonckt, J. M.: On the problems of selecting interaction objects. In: G. Cockton, S. Draper & G. Weir (eds.): People and Computers IX. Cambridge: Cambridge University Press 1994, pp. 163-178.
3. Brunner, M. and M. Rauterberg: Hierarchische oder netzartige Dialogstruktur bei multimedialen Informationsystemen: eine experimentelle Vergleichsstudie. Technical Report MM-2-93. Institut für Arbeitspsychologie, Zürich: Eidgenössische Technische Hochschule (1993).

4. Denert, E.: Specification and design of dialogue systems with state diagrams. In: E. Morlet and D. Ribbens (eds.): International Computing Symposium '77. Amsterdam: North-Holland 1977, pp. 417-424.
5. Fisher, J.: Defining the novice user. Behaviour and Information Technology 10(5), 437-441 (1991).
6. Dix, A., J. Finlay, G.: Abowd and R. Beale: Human-Computer Interaction. New York: Prentice Hall (1993).
7. Grützmacher, B.: Datenpräsentation und Lösungsverhalten in einer komplexen, simulierten Problemsituation. Unpublished Master Thesis. (Philosophische Fakultät I, Psychologisches Institut, Abteilung Angewandte Psychologie). Zürich: Universität Zürich (1988).
8. IFIP: Report of the 1st Meeting of the European User Environment Subgroup of IFIP WF 6.5. German National Center for Computer Science (GMD), P.O. 1316, D-5202 Sankt Augustin (Germany) (1981).
9. Jeffries, R. and H. Desurvire: Usability testing vs. heuristic evaluation: was there a contest? SIGCHI Bulletin 24(4), 39-41 (1992).
10. Karat, J.: Software Evaluation Methodologies. In: M. Helander (ed.): Handbook of Human-Computer Interaction. Amsterdam: Elsevier 1988, pp. 891-903.
11. Kirakowski, J. and M. Corbett: Effective Methodology for the Study of HCI. In: H. Bullinger and P. Polson (eds.): Human Factors in Information Technology vol. 5. Amsterdam: North-Holland (1990).
12. Kleinbaum, D. and Kupper, L.: Applied Regression Analysis and other Multivariate Methods. Belmont: Wadsworth (1978).
13. Laverson, A., K. Norman and B. Shneiderman: An evaluation of jump-ahead technique in menu selection. Behaviour and Information Technology 6(2), 97-108, (1987).
14. Paap, K. and R. Roske-Hofstrand: Design of menus. In: M. Helander (ed.): Handbook of Human-Computer Interaction. Amsterdam: Elsevier 1988, pp. 205-235.
15. Rauterberg, M.: An empirical comparison of menu-selection (CUI) and desktop (GUI) computer programs carried out by beginners and experts. Behaviour and Information Technology 11(4), 227-236 (1992).
16. Rauterberg, M.: Quantitative Measures to Evaluate Human-Computer Interfaces. In: M. Smith and G. Salvendy (eds.): Human-Computer Interaction: Applications and Case Studies. Amsterdam: Elsevier 1993 (Advances in Human Factors/ Ergonomics vol. 19A, pp. 612-617).
17. Rengger, R.: Indicators of usability based on performance. In: H-J. Bullinger (ed.): Human Aspects in Computing: Design and Use of Interactive Systems with Terminals. Amsterdam: Elsevier 1991, pp. 656-660.
18. Shneiderman, B.: Designing the user-interface. Amsterdam: Addison-Wesley (1987).
19. Ulich, E., M. Rauterberg, T. Moll, T. Greutmann and O. Strohm: Task orientation and user-oriented dialog design. International Journal of Human-Computer Interaction 3(2), 117-144 (1991).
20. Wirth, N. and J. Gutknecht: Project Oberon - The design of an operating system and compiler. Reading (MA): Addison-Wesley (1992).