in: Human-Computer Interaction: Tasks and Organisation. (G. van der Veer, M. Tauber, S. Bagnara & M. Antalovits, eds.) Roma: CUD Publ., 1992, pp. 295-307.

A Method of a Quantitative Measurement of Cognitive Complexity

M. Rauterberg

Work- and Organizational Psychology Unit
Swiss Federal Institute of Technology (ETH), Zurich, Switzerland

ABSTRACT

A theoretical framework to conceptualize measure of behaviour complexity (BC), system complexity (SC) and task complexity (TC) was developed. From this framework cognitive complexity (CC) is derived as CC = SC + TC - BC. In an empirical study to investigate different measures of cognitive complexity 6 beginners and 6 experts solved 4 different tasks with an interactive database management system. To analyze the empirical data recorded during the interactive sessions a special program was developed. The program's purpose and the program architecture are described. 4 different approaches from the literature to measuring complexity in a quantitative way were considered and discussed. Application of these 4 approaches were compared and tested against the empirical results of the experiment. The measure of McCabe (1976) proved to be the most effective.

1. Introduction

This study was carried out to support the formal analysis of studying user keystroke behaviour. The normal design cycle used to construct a formal model is a top down approach. In this paper we present an automatic bottom up approach to construct a formal description of user behaviour. The formalism we have selected to model the user's knowledge with finite state/transition systems, is the Petri net theory.
There are different formalisms for constructing user models the context of human computer interaction; TAG (Payne & Green 1986), CLG (Moran 1981), GOMS (Card, Moran & Newell 1983), CCT (Kieras & Polson 1985), and different kinds of grammars BNF (Reisner 1981), EBNF (Reisner 1984), etc. Using any of these formalisms the investigator is obliged to design the pure (="error free") user model in a top down approach. Then he or she can try to prove the model with "error free" empirical data. This is difficult, expensive and insufficient, and one of the consequences is that most of the formal models exist only as paper versions and have not been implemented as executable simulations. For a more detailed critique of the mentioned formalism see Greif & Gediga (1987), Sutcliffe (1989), Karat & Bennett (1991) or Benyon (1992).
If user models can be constructed using an automatic, bottom up approach, then the handling of formal models becomes easy. To achieve this, we first of all develop a theoretical framework, which fits our empirical approach. Based on this framework we are able to test four different measures of complexity according to their "discriminating power".

2. Cognitive complexity in man computer interaction

Cognitive complexity has been defined as "an aspect of a person's cognitive functioning which at one end is defined by the use of many constructs with many relationships to one another (complexity) and at the other end by the use of few constructs with limited relationships to one another (simplicity)" (Pervin 1984, p. 507). Transferring this broad definition to the man computer interaction could mean: the complexity of the user's mental model of the dialog system is given by the number of known dialog contexts ("constructs") on one hand, and by the number of known dialog operations ("relationships") on the other hand. The scope of this paper does not include approaches based on questionnaires (Scott, Osgood & Peterson 1979, McDaniel & Lawrence 1990). Comparing the traditional approaches to measure cognitive complexity Grote and James (1990, 419-420) come to the following conclusion, "it appears that the most basic problem concerning the concept of cognitive complexity is its lack of an unambiguous and uniformly used definition and operationalization".

2.1. Definition of cognitive complexity

Observing the behaviour of people solving a specific problem or task is our basis for estimating "cognitive complexity (CC)". The cognitive structures of users are not direct observable, so we need a method and a theory to use the observable behaviour as one parameter to estimate CC. A second parameter is a description of the action or problem solving space itself. The third parameter is an "objective" measure of the task or problem structure.

We call the complexity of the observable behaviour the "behaviour complexity (BC)". This behaviour complexity can be estimated by analysing the recorded concrete task solving process, which leads to an appropriate task solving solution. The complexity of a given tool (e.g. an interactive system) we call "system complexity (SC)". The last parameter we need is an estimation of the "task complexity (TC)".
The necessary task solving knowledge for a given task is constant. This knowledge embedded in the cognitive structure (CC) can be observed and measured with BC. If the cognitive structure is too simple, then the concrete task solving process must be filled up with a lot of heuristics or trial and error strategies. Learning how to solve a specific task with a given system means that BC decreases (to a minimum = TC) and CC increases (to a maximum = SC). We assume, that the difference (BC–TC) is equal to the difference (SC–CC).
To solve a task, a person needs knowledge about the dialogue structure of the interactive software (measured by SC) and about the task structure (measured by TC). SC is an upper limit for TC (SC≥TC); this aspect means, that the system structure constrains the complexity of the observable task solving space. We can now state with the constraints (BC≥TC) and (SC≥CC), that:

$$BC - TC = SC - CC \qquad \text{(Formula 1)}$$

Transforming this Formula 1 to get CC alone, results in Formula 2:

$$CC = SC + TC - BC \qquad \text{(Formula 2)}$$

The parameter SC is given by the concrete system structure, so we have to apply a given complexity measure to this system structure. The parameter TC can be estimated either in a theoretical or in an empirical way. All apriori descriptions of task structures are usable to calculate a theoretical TC. If we have empirical data of different task solving processes of different persons, then we can estimate TC using the minimum of all observed BCs per task. Given a sample of different complete task solving processes, the best approximation for TC seems to be the minimal solution regarding a specific complexity measurement. One plausible consequence of this assumption is that CC is equal to SC in the case of "best solution" (TC=BC). This is the approach we are presenting here.

2.2. Quantitative measurements of complexity

The cognitive complexity (CC) is defined by Formula 2. To measure the system complexity (SC) we need a description of the interactive system "behaviour". In the context of this work we are using a state/transition matrix to describe all possible dialog actions the user can use to change from one dialog state to another. Then we need to carry out an empirical investigation to observe and record user behaviour solving a set of tasks with the interactive system.
To measure complexity we introduce four different metrics. The first simple metric we use is given by Stevens, Myers and Constantine (1974). They count the number of dialog states (S: states) to measure the "absolute structural complexity of a net" (see Formula 3).

$$C_{state} = S \qquad \text{(Formula 3)}$$

The "relative structural complexity" of a net is the ratio of the number of connections between dialog states (T: transition) to the total number of dialog states (S: state) (see Formula 4). The measure C_{fan} represents the average number of connections per state, so we call this complexity measure the "fan degree" of the net.

$$C_{fan} = T / S \qquad \text{(Formula 4)}$$

The third metric we use is published by McCabe (1976). His complexity measure was created in the context of software development, to analyse large programs. The mathematical background of this measure is graph theory, which offers several procedures to describe different aspects of net structures. If we have two dimensional net structures, then we can calculate the number of basic cycles ("holes") in the net with this complexity measure (see Formula 7). The complexity is defined by the difference of the total number of connections (T: transition) and the total number of states (S: state). The parameter P is a constant to correct the result of Formula 5 in the case of a sequence (S = T + P); the value of P in our context is 1. The semantic of C_{cycle} can be described by the number of "holes" in a net. C_{cycle} is a metric to calculate the number of linear independent cycles of a plane and coherent net.

$$C_{cycle} = T - S + P \qquad \text{(Formula 5)}$$

The fourth metric is given by Kornwachs (1987). His concept was developed for a general system with elements and connections. His complexity measure was explicitly designed for man-computer interaction. The idea of this measure is to estimate the actual net density compared to the maximal possible net density.

The maximal possible net density increases proportional to the square of the number of states. Let S be the number of all elements ("states") in a given system I; the matrix of all realized connections c is given by: $c_{ij}=0$, if element $i \in I$ is disconnected with element $j \in I$, and $c_{ij}=1$, if element $i \in I$ is connected with element $j \in I$. The number of all realized (= really existing) connections is $t= \Sigma\Sigma\ c_{ij}$. All possible directed connections s of the system I can be calculated by s=S(S-1). The structural density d is given by d=t/s. The "structural degree of complexity" Cdensity is now defined as the structural density d.

$$C_{density} = T / (S*(S-1)) \qquad \text{(Formula 6)}$$

With Cstate, Cfan, Ccycle, and Cdensity we have four different metrics to measure complexity. We shall discuss the advantages and disadvantages of these four quantitative metrics in the context of an empirical investigation below.

2.3. Representations of cognitive complexity

Measurement of complexity of a system described with a state/transition matrix in a quantitative way is one central issue; the other central issue is to transform the structure of a given system in an "appropriate form". One qualitative approach to figure complexity is drawing the "net structure" of the system. If we use Petri-Nets (Petri 1980) instead of the equivalent state/transistion formalism (Wasserman 1985), we can simulate the user's mental model in an executable form with Petri-Net simulators.
A Petri-Net is a mathematical structure consisting of two non-empty disjoint sets of nodes, called S-elements and T-elements, respectively, and a binary relation F, called the flow relation. F connects only nodes of different types and leaves no node isolated. Nets can be interpretated by using a suitable pair of concepts for the sets S (signified by curved brackets "( )") and T (signified by square brackets "[ ]") and a suitable interpretation for the flow relation F (signified by an arrow "->"). The means/activity interpretation allows one to describe the static structure of a system with several active and passive functional components: means (S) = real or informational entity, and activity [T] = (repeatable) action of a system. The flow relation F means: [a] -> (m), the activity a (e.g. a dialog operation) produces means m (e.g. a dialog state); (m) -> [a], activity a uses means m. The main operations (relations) between two nets are abstraction, embedding and folding (Genrich, Lautenbach & Thiagarajan 1980). Folding is the most important operation in our context.

3. The Automatic Mental Model Evaluator (AMME)

What is the main concern of a user interacting with a software system? The user must build up a mental representation of the system's structure and gain knowledge about the functions of this system with respect to a set of tasks. Furthermore, he must learn the language, i.e., a set of symbols, their syntax, and operations connected to them, to evoke interaction sequences (the interactive "processes") related to task and subtask functions. So, the user's representations of the system structure are models of a virtual machine. A "virtual machine" is defined as a representation of the functionality of a system (functional units and their behaviour). The most important point for the user is the relation between task and machine, and not so much the internal structure of the machine's system. Consequentely, the task for the human factors engineer is to model a suitable interface as a representation of the virtual machine which can serve as a possible mental representation for the user.
The symbolic representation of the machine system consists of the following elements: 1. objects (things to operate on), 2. operations (symbols and their syntax), and 3. states (the "dialog states"). The mental model of the user can be structered in representing objects, operations, states, system structure, and task structure.

3.1. The "idea" of AMME

If a user interacts with a dialog system, he or she produces a sequence of states and transitions (s') -> [t'] -> (s") -> [t"] -> (s') -> [t'] -> (s") -> [t"] -> ... Each state corresponds to a dialog context, and each transition corresponds to a dialog operation. This sequence is called a "process". Measurable facts in the process are for example number of states and transitions, time per transition, etc. This measurements can be easily done based on a protocol of the user's behaviour automatically recorded by the dialog system in a "logfile" (the logfile recording technique; Crellin, Horn & Preece 1990).
To measure the complexity of the mental model which generates the actual process, we first need a mapping procedure from the observable process to the embedded structure of this process. This mapping procedure can be done with the folding operation in the context of Petri nets. Folding a process means to map S-elements onto S-elements and T-elements onto T-elements while keeping the F-structure. The result is the structure of the performance net. The result of a folding operation of our example sequence above is a loop (s') -> [t'] -> (s") -> [t"] -> {back to (s')}. This simple loop with two different states and two different transitions is the whole net structure we need to produce the process given above.

The prime idea of our approach is based on the actual observation of users performing a specific task. The key to the interpretation of the protocols is a map of the complete task solving domain, on which the behaviour of individual processes is drawn. This task solving domain in our approach is the whole dialog net structure of the interactive system. A sequence of keystrokes can be contemplated as a sentence derived from a defined grammar or as a process derived from a Petri net. The state transition net, as a complete description of the software the user is interacting with, can be used to identify the equal states in the keystroke sequence. All parts of the user's keystroke sequence between two dialog states are elementary processes. All elementary processes can be combined to form a Petri net (the "folding" operator). The "folded" Petri net is a formal description ("model") of the procedural knowledge of the users behaviour.

3.2. The program structure of AMME

The tool AMME consist of four different programms: (1) the dialog system with the logfile recording feature (ADIMENS 1988); (2) a transformation program, which translate the binary logfile in a readable ascii file; (3) the analysing program PACEGEN (Hofmann & Rudnik 1991), which extracts the net of the task and user specific process and calculates different quantitative aspects of the generated net; (4) the Petri net simulator PACE (Dähler 1989), which reads the extracted net given in a special syntactical format. The analysing program PACEGEN version 1.0 is programmed in MacMETH-Modula on Macintosh computers. The Petri net simulator runs in the SmallTalk runtime environment on Macintosh computers.

3.3. Operating and usage of AMME

First, we shall describe the dialog structure of ADIMENS 2.21. This description is written as an ascii file in a special state/transition syntax, defined by Hofmann and Rudnik (1991). This dialog system description is one of the neccessary inputs to the analysing program PACEGEN. The other input to PACEGEN is the logfile after translation in an ascii format. PACEGEN produce two output files: a protocol file with different quantitative measures of the extracted net and a net description file in a readable form for PACE.
The generation of the file with the dialog description is an iterative procedure. With each new logfile analysed by PACEGEN in most cases the description file must be updated. After analysing the whole set of logfiles in the first trial all logfiles must be reanalysed in a second trial. This procedure guaranties, that the calculation of the quantitative measures of each generated net is correct.

3.4. Limitations of AMME

The actual version of PACEGEN is restricted to logfiles only generated by ADIMENS 2.21. In the next version of PACEGEN the syntax of the logfiles could be defined by the researcher given a special description language to express the specific syntactical structure of the logfiles. The more general restriction comes from the type of the dialog structure (the "task domain") itself: only dialog structures, which can be expressed with a context-free grammar, are describable.

4. Analyzing user behaviour recorded in logfiles

4.1. The empirical investigation

We tested the dialog system ADIMENS 2.21 with 12 users (Ulich et al. 1991). These 12 users had have to solve 10 different benchmark tasks in the context of operating the data base system ADIMENS. We present the results of the first four different benchmark tasks.

4.1.1. Subjects

Two different user groups took part in this experiment: a beginner group (N=6: 4 women, 2 men; average age of 27 years), and an expert group (N=6: 0 woman, 6 men; average age of 38 years).

4.1.2. Experimental setting

First, the experience with computers was measured with a 115-item questionnaire and with interviews, then the beginner group was instructed for 1.5 hours in handling the database system. The expert group had 1,740 hours of experience in handling the database system ADIMENS. Their total computer experience of about 7,500 hours was the result of their daily work using different types of computers and software systems. The duration of the actual task solving session was about 30 minutes. Each keystroke with a timestamp was recorded in a logfile. Each user needed about 2 hours for the whole experiment (10 tasks, individual sessions).

4.1.3. System description

The dialog system was the relational data base system ADIMENS version 2.21 with a character oriented user interface (CUI) running on standard IBM PC's with standard keyboard. The whole dialog structure is strictly

hierarchical organized with three levels: (1) the main menu has 7 dialog operations (ordinary ascii characters chosen from a menu) to go down to 7 different modules, and 5 function keys with specific semantics; (2) at the module level each module has exactly 4 different dialog operations to change to routines and on average 4.1 (±1.7; range: 0-5) function keys with specific semantics; (3) at the routine level the user has only on average 3.7 (±2.9; range: 0-10) different function keys to control the dialog (additionally all ascii keys and the 4 cursor keys are usable).
The number of all ordinary dialog contexts (main menu, modules, routines) is 1+7*4=29. But to describe the complete dialog structure with all help, error and additional dialog states we need at least 144 different states. To change from one state to the other the system offers overall 358 different dialog operations (transitions).

4.1.4. Task description

In the experiment all users had to play the role of a camping place manager. This manager uses a database system with a data base consisting of three data files: PLACE, GROUP, and ADDRESS. All users had to solve the following four different tasks operating the database system:

Task 1: "How many data records are in the file ADDRESS, in the file PLACE, and in the file GROUP? Find out, please."

The user has to activate a specific menu option ("Datafile" in module "Info" of the menu interface) and to read the file size (solutions: PLACE = 17 data records, GROUP = 27 data records, ADDRESS = 280 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, GROUP, ADDRESS).

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:	Germany
Place number:	07
Remarks:	Database system dealer can give a demonstration."

The user must select a certain data record (file: ADDRESS), update the data record with regard to the three attributes: State, Place number, Remarks.

Task 4: "Define a filter for the file PLACE with the following condition: all holidaymakers arrived on date 02/07/87. Apply this filter to the file PLACE, and show the content of all selected data records in the mask browsing mode on the screen."

The user must define a filter for the attribute "arrival date", apply the filter to the data file PLACE, and display the content of each data record found on the screen.

4.1.5. Dependent measures

One of the most important dependent measures is the task solving time. This variable and the results of the computer experience questionnaire are needed for validating the different complexity measures. With the analysing program PACEGEN we obtained the following measures per user and per task: the number of all different transitions (T vector: "# of transitions") and the number of all different dialog states (S vector: "# of states").
We do not assume a massive learning process during the task solving period. So, a good measure of CC must differentiate between beginners and experts, but not overall between the four tasks.

4.2. Experimental results

Let us begin with the performance measure "task solving time". The means of the two groups are different (see Fig. 1). The experts are significantly faster than the beginners (see Tab. 1). This expected result is fundamental for validating the complexity measures. The other important results are the significant differences between the four tasks. [Two beginners were not be able to solve task 4, so there are two missing data. These missing data are estimated by group means and replaced.]
If the performance measure is a coarse estimation of complexity, then the order of the four tasks according to their complexity is: lowest is task 1 ("info"), followed by task 4 ("filter"), task 2 ("delete") and task 3 ("edit"). A close look at task 3 shows, that the user needs the knowledge to handle 15 different states just for the editor required in the dialog context of the routine "update"; to change from one editor state to another the user needs at least 45 different transitions (e.g. different semantics of the cursor keys). In the following analysis we evaluate the logfiles only on the routine level, so we are not able to measure the complexity below the routine level.

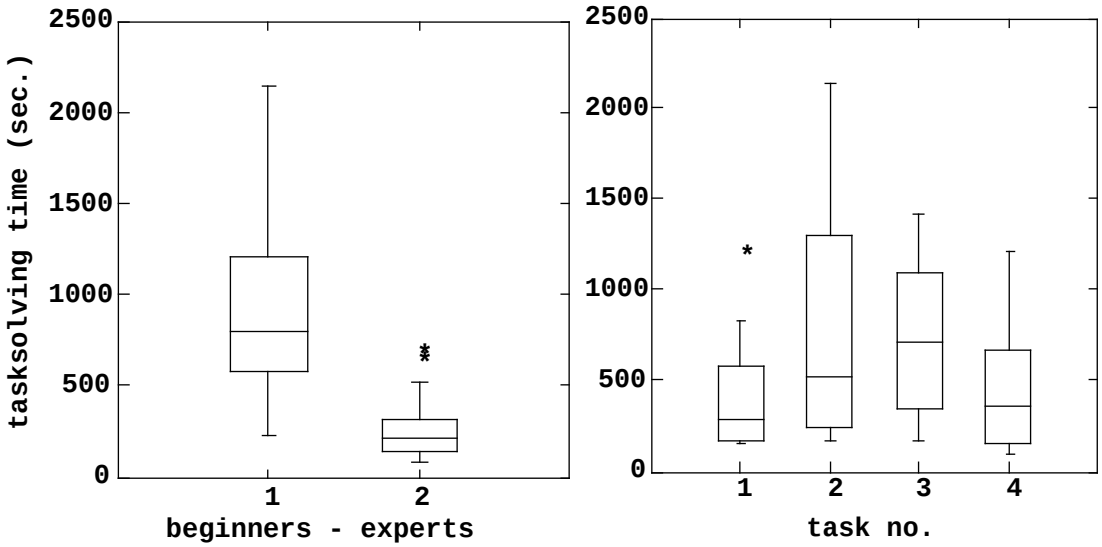

Figure 1 These two Box-and-Whisker Plots show the task solving time seperated for beginners-experts and tasks 1-4 (N=48). The central box of each condition covers the middle 50% of the data values, between the lower and upper quartiles. The "whiskers" extend out to the extremes, while the central line is at the median.

Table 1 Analysis of variance of the variable "task solving time" (N=48).

SOURCE	SUM-OF-SQUARES	DF	MEAN-SQUARE	F-RATIO	P
experience	4987786.021	1	4987786.021	45.283	0.001
tasks	1326184.229	3	442061.410	4.013	0.014
exp. x tasks	489222.229	3	163074.076	1.481	0.234
ERROR	4405844.500	40	110146.113		

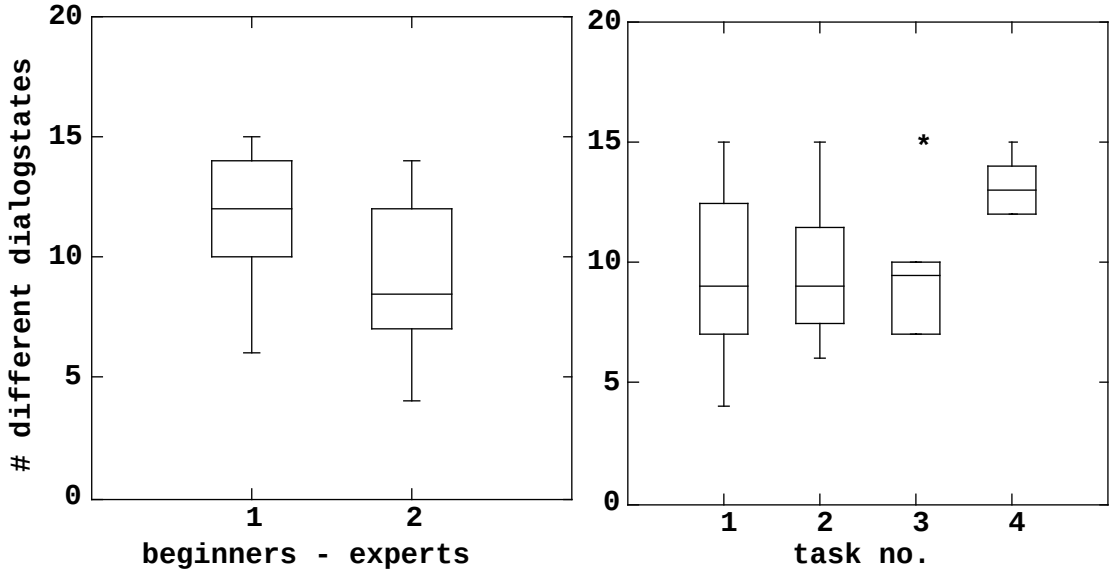

Figure 2 These two Box-and-Whiskers Plots show the results of the "number of different dialog states" (N=45, 3 outliers excluded).

Table 2 Analysis of variance of the "number of different dialog states" (N=45, 3 outliers excluded).

SOURCE	SUM-OF-SQUARES	DF	MEAN-SQUARE	F-RATIO	P
experience	54.028	1	54.028	8.830	0.005
tasks	106.271	3	35.424	5.789	0.002
exp. x tasks	15.260	3	5.087	0.831	0.485
ERROR	226.400	37	6.119		

The next estimation of complexity is the number of different dialog states. We need 144 different dialog states to describe the dialog structure of the interactive system. Solving a given task the user is navigating through this dialog structure. The average number of different dialog states the beginners need to solve all tasks is significantly higher than the average of the expert group (see Fig. 2, and Tab. 2). This result can be easily explained by the different experience of operating the system of these groups: the beginners need more heuristic search strategies to solve the tasks than the experts. There is also a significant difference between task 4 and the other three tasks.

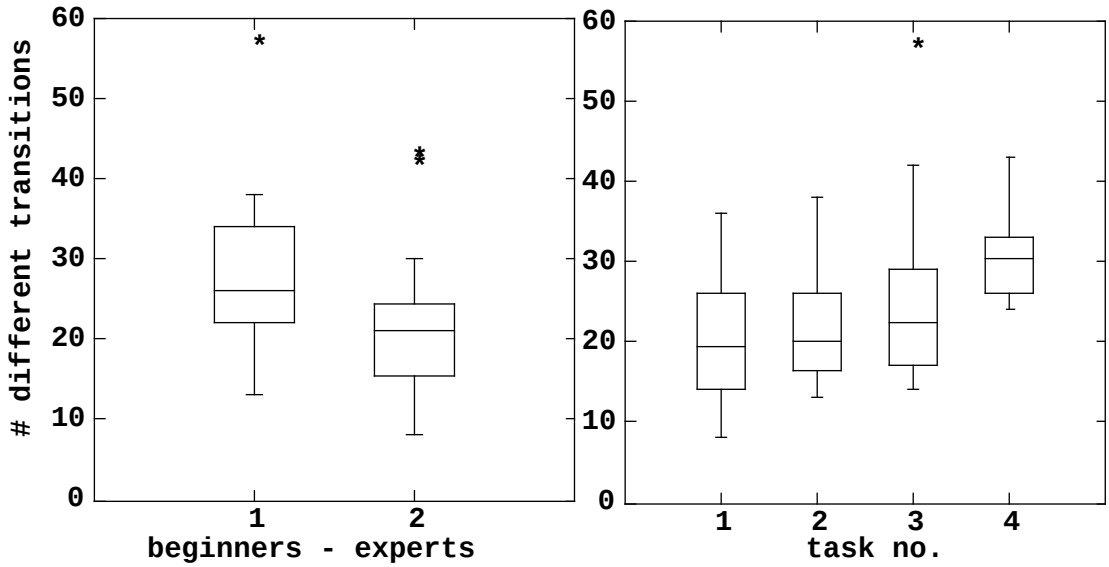

Figure 3 These two Box-and-Whiskers Plots show the results of the "number of different transitions" (N=46, 2 outliers excluded).

Table 3 Analysis of variance of the "number of different transitions" (N=46, 2 outliers excluded).

SOURCE	SUM-OF-SQUARES	DF	MEAN-SQUARE	F-RATIO	P
experience	508.255	1	508.255	6.682	0.014
tasks	654.711	3	218.237	2.869	0.049
exp. x tasks	88.974	3	29.658	0.390	0.761
ERROR	2890.500	38	76.066		

The beginners also use significantly more different transitions than the experts (see Fig. 3, and Tab. 3). This result support the interpretation, that the beginners need more heuristics to solve the tasks than the experts. In solving task 4 all users need the most transitions. The main effect "tasks" in the analysis of variance is significant at the 5% level (see Tab. 3).
These results of the three dependent measures are the basis for validating the 4 different complexity measures.

5. Validation of the four different complexity measures

To estimate the cognitive complexity we must first calculate the behaviour complexity ("BC") of each user and each task. Then we estimate the task complexity ("TC") of each task by searching for the minimum of the 12 empirical values of the behaviour complexity (the "best" solution). The system complexity is given by the system description, and is always constant in the context of a specific complexity measure.

5.1. "Cognitive Complexity" based on the measure of "different dialog states"

,First we present the results of the measure of cognitve complexity CC_{state} of Stevens et al. (1974) according to the number of different dialog states (S: states). The results for the behavioral complexity BC_{state} are shown above (see Fig. 2, and Tab. 2). The following estimations of the behavioural, system, and task complexity are set:

Behaviour Complexity:	$BC_{state} = S$	
System Complexity:	$SC_{state} = 144$ (number of total states)	
Task Complexity:	TC_{state}:	$\min[BC_{state}]_{task\text{-}1} = 4$,
		$\min[BC_{state}]_{task\text{-}2} = 6$,
		$\min[BC_{state}]_{task\text{-}3} = 7$,
		$\min[BC_{state}]_{task\text{-}4} = 12$
Cognitive Complexity:	$CC_{state} = SC_{state} + TC_{state} - BC_{state}$	

The global equation to calculate the cognitive complexity CC is given by Formula 2 above. The TCs are in the expected direction: task 1 low complex, task 4 high complex.

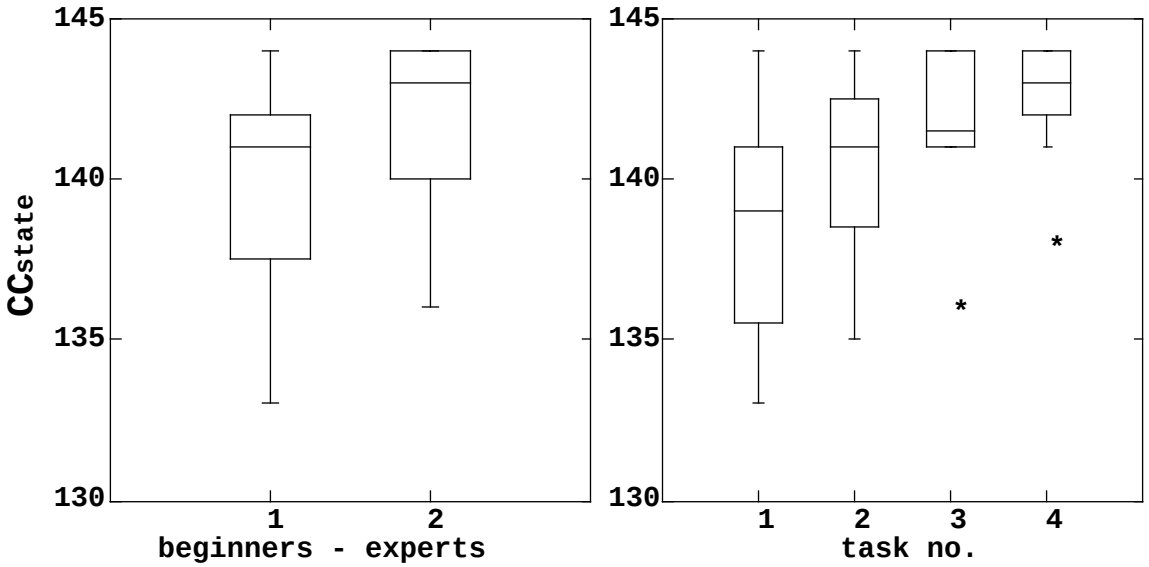

Figure 4 These two Box-and-Whiskers Plots show the results of the cognitive complexity "CC_{state}" measure according to Stevens et al. (N=46, 2 outliers excluded).

Table 4 Analysis of variance of the cognitive complexity "CC_{state}" measure according to Stevens et al. (N=46, 2 outliers excluded).

SOURCE	SUM-OF-SQUARES	DF	MEAN-SQUARE	F-RATIO	P
experience	44.579	1	44.579	6.757	0.013
tasks	107.853	3	35.951	5.449	0.003
exp. x tasks	24.230	3	8.077	1.224	0.314
ERROR	250.700	38	6.597		

This measure CC_{state} is good enough to differentiate between beginners and experts (see Fig. 4, and Tab. 4). As one can see in Figure 4, the cognitive complexity CC_{state} of the performance model of the experts is significantly higher than the cognitive complexity CC_{state} of the beginners. This outcome is valid and meaningful. But the result, that the average complexity CC_{state} of the cognitive model of task 3 and 4 is higher than the complexity CC_{state} of task 1 and 2, is counter intuitive. This result would enforce the interpretation, that the users had more knowledge of the "more complex" tasks than of the "less complex" tasks. This outcome can only be explained, if there was a massive learning process during the task solving period. This assumption could be plausible for beginners, but not for the experts, so, if this interpretation is correct, then we must have a significant interaction "exp. x tasks" in the analysis of variance. But there is no significant interaction (see Tab. 4). Let us see, how this aspect will be handled by the other 3 measures of complexity.

5.2. "Cognitive Complexity" based on the measure of "fan degrees"

Now we present the results of the measure of cognitve complexity of Stevens et al. (1974) according to the average fan degree of each dialog state. The total number of all possible dialog states (S) is 144, and of all transitions (T) is 358 (given by the system description).

Behaviour Complexity: $BC_{fan} = T/S$

System Complexity: $SC_{fan} = 358/144 = 2.486$

Task Complexity: TC_{fan}: $\min[BC_{fan}]_{task\text{-}1} = 1.875$,
$\min[BC_{fan}]_{task\text{-}2} = 2.077$,
$\min[BC_{fan}]_{task\text{-}3} = 2.000$,
$\min[BC_{fan}]_{task\text{-}4} = 2.000$

Cognitive Complexity: $CC_{fan} = SC_{fan} + TC_{fan} - BC_{fan}$

The average fan degree of all 144 dialog states is 2.486 (SC_{fan}). This means that the user can choose on average between 2 and 3 alternatives to go on navigating in the dialog structure. If we take the fact that the 29 main dialog contexts described above have between 3.7 and 12.0 transitions alternatives (see section on "system description"), then we can derive from the system description, that the most dialog states in the system description are of simple decision quality: "go on" ($SC_{fan} = 1$; 48%), "yes" or "no", "ok" or "cancel" ($SC_{fan} = 2$; 22%), ($SC_{fan} = 3$; 11%) and ($SC_{fan} > 3$; 19%). The TC_{fan} measure does not differentiate very well between the four tasks.

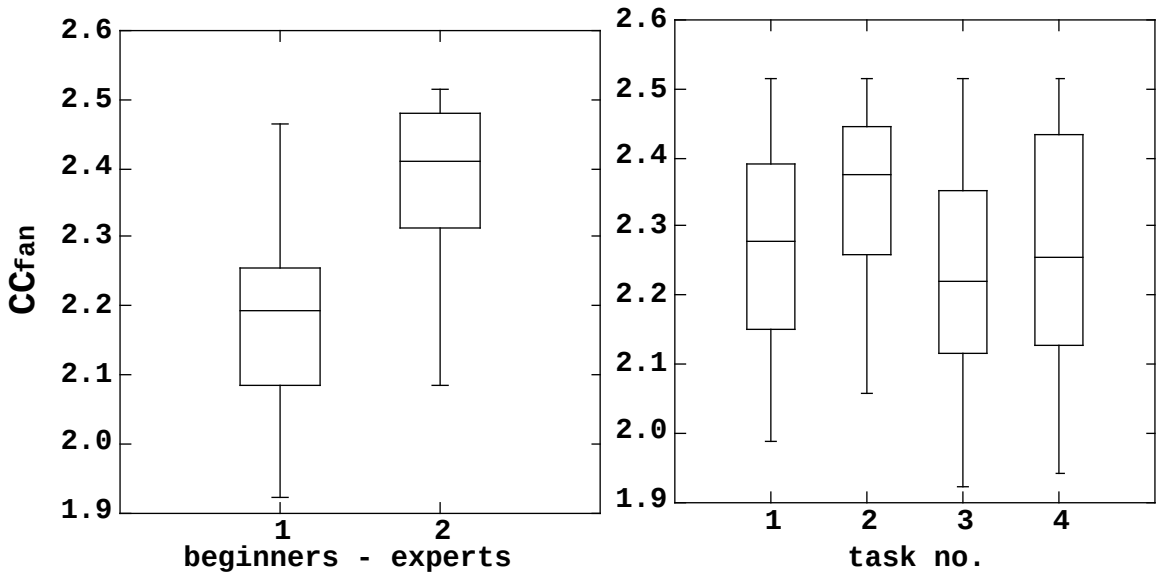

Figure 5: These two Box-and-Whiskers Plots show the results of the cognitive complexity "CC_{fan}" measure according to Stevens et al. (N=48)

Table 5 Analysis of variance of the cognitive complexity "CC_{fan}" measure according to Stevens et al. (N=48).

SOURCE	SUM-OF-SQUARES	DF	MEAN-SQUARE	F-RATIO	P
experience	0.494	1	0.494	29.026	0.001
tasks	0.085	3	0.028	1.667	0.189
exp. x tasks	0.022	3	0.007	0.423	0.738
ERROR	0.680	40	0.017		

This measure CC_{fan} is able to differentiate between beginners and experts in the expected direction: the complexity of the cognitive models of the experts is significantly higher than the complexity of the beginner models (see Fig. 5, and Tab. 5). The average cognitive complexity CC_{fan} of all users does not differ in the four tasks.

5.3. "Cognitive Complexity" based on the measure of "net cycles" (McCabe)

The empirical values of the number of states (S) and transitions (T) of each user per task are given by the vectors S and T.

Behaviour Complexity: $BC_{cycle} = T - S + 1$

System Complexity: $SC_{cycle} = 358 - 144 + 1 = 215$

Task Complexity: TC_{cycle}: $\min[BC_{cycle}]_{task\text{-}1} = 5$, $\min[BC_{cycle}]_{task\text{-}2} = 8$, $\min[BC_{cycle}]_{task\text{-}3} = 8$, $\min[BC_{cycle}]_{task\text{-}4} = 13$

Cognitive Complexity: $CC_{cycle} = SC_{cycle} + TC_{cycle} - BC_{cycle}$

It is important to notice, that the task complexity TC_{cycle} of the different tasks is in the expected direction: task 1 low in complexity and task 4 is highly complex.

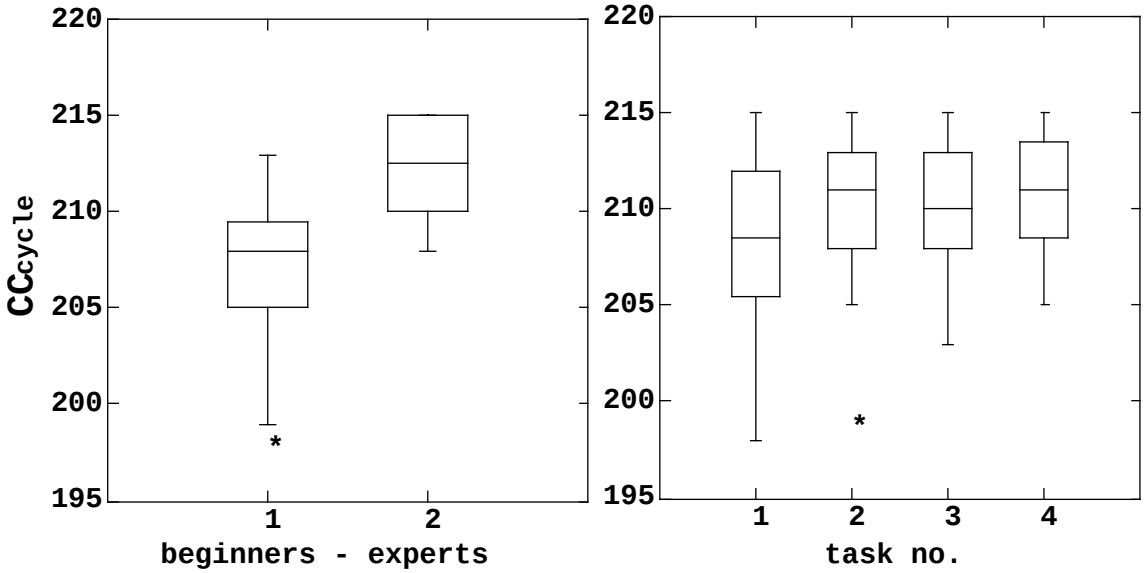

Figure 6: These two Box-and-Whiskers Plots show the results of the cognitive complexity "CC_{cycle}" measure according to McCabe (N=45; 3 outliers excluded).

Table 6 Analysis of variance of the cognitive complexity "CC_{cycle}" measure according to McCabe (N=45, 3 outliers excluded).

SOURCE	SUM-OF-SQUARES	DF	MEAN-SQUARE	F-RATIO	P
experience	272.596	1	272.596	21.547	0.001
tasks	53.326	3	17.775	1.405	0.257
exp. x tasks	10.813	3	3.604	0.285	0.836
ERROR	468.100	37	12.651		

The cognitive complexity CC_{cycle} of the mental models of beginners is significantly less complex than the cognitive complexity of the mental models of the experts. This result is in accordance with our starting point. CC_{cycle} does not differ in the four tasks (see Fig. 6); so, we do not find a significant difference for the factor "tasks" in the analysis of variance (see Tab. 6). The maximum of CC_{cycle} is SC_{cycle} (= 215). This maximum is reached by the expert group (see Fig. 6).

5.4. "Cognitive Complexity" based on the measure of "net density" (Kornwachs)

The empirical values of the number of states (S) and transitions (T) of each user per task are given by the vectors S and T.

Behaviour Complexity: $BC_{density} = T/(S*(S-1))$

System Complexity: $SC_{density} = 358/(144*(144-1)) = 0.017$

Task Complexity: $TC_{density}$: $\min[BC_{density}]_{task\text{-}1} = 0.667$,
$\min[BC_{density}]_{task\text{-}2} = 0.433$,
$\min[BC_{density}]_{task\text{-}3} = 0.333$,
$\min[BC_{density}]_{task\text{-}4} = 0.182$

Cognitive Complexity: $CC_{density} = SC_{density} + TC_{density} - BC_{density}$

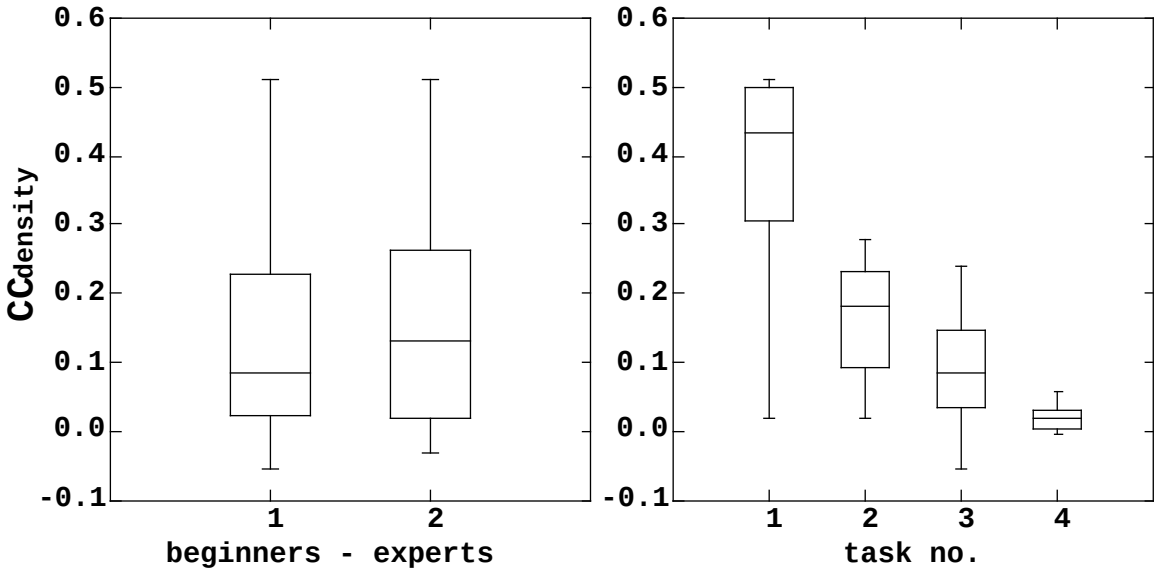

Figure 7: The two Box-and-Whiskers Plots show the results of the cognitive complexity "$CC_{density}$" measure according to Kornwachs (N=48)

Table 7 Analysis of variance of the cognitive complexity "$CC_{density}$" measure according to Kornwachs (N=48).

SOURCE	SUM-OF-SQUARES	DF	MEAN-SQUARE	F-RATIO	P
experience	0.042	1	0.042	4.355	0.043
tasks	0.916	3	0.305	31.621	0.001
exp. x tasks	0.042	3	0.014	1.438	0.246
ERROR	0.386	40	0.010		

The estimations of task complexity $TC_{density}$ are counter intuitive: task 1 is highly complex and task 4 is low in complexity! Also, the system complexity $SC_{density}$ is lower than each task complexity $TC_{density}$. This fact leads to negative complexity values (see Fig. 7). We did not include in the construction of $CC_{density}$ the aspect of the structural complexity of hierarchies (see Kornwachs 1987), so, perhaps, this surprising result comes from excluding this aspect.

The measure $CC_{density}$ differentiates poorly between beginners and experts, but very well between the four different tasks (see Tab. 7). The average values of $CC_{density}$ of the cognitive performance models according to the tasks are in a surprising direction: the users loose their knowledge during the task solving period (see Fig. 7). This outcome is not plausible.

6. Discussion and conclusions

Based on the assumption, that our theoretical model of cognitive complexity given by Formulas 1-4 is valid and meaningful, we are able to test and validate the four different measures of complexity. After the first test trial presented above, we bring the four different complexity measures (CCs) in relation to other aspects of our experimental "reality". First, we calculate the product moment correlation of the CCs with "task solving time". As one can see from Table 8, there is a significant negative correlation between CC_{state}, CC_{fan}, and CC_{cycle} with the task solving time. This is a valid and plausible outcome. Only $CC_{density}$ correlates positively (insignificant) with task solving time (see Tab. 8) and with number of dialog states (see Tab. 9), so we can exclude the measure $CC_{density}$ from further considerations.

Table 8 Product moment correlation matrix of "task solving time" with the different quantitative measures (N=48).

	CORRELATION	PROBABILITIES
# of states	0.356	0.013
# of transistions	0.461	0.001
CCstate	-0.427	0.002
CCfan	-0.576	0.001
CCcycle	-0.576	0.001
CCdensity	0.126	0.394

The measues CC_{state}, CC_{fan}, and CC_{cycle} are highly intercorrelated (see Tab. 9), except for CC_{state} and CC_{fan}. So we can assume that both measures estimate different qualities of a net structure. This result sounds plausible. If we take into account that CC_{state} differs with the tasks (see Fig. 4 and Tab. 4), then we can exclude the measure CC_{state} from further studies, as well.

Table 9 Product moment correlation matrix of the two quantitative net aspects and the four measures of cognitive complexity (N=48).

	#STATES	#TRANSITION	CCstate	CCfan	CCcycle
CCstate	-0.685	-0.675			
CCfan	-0.363	-0.527	0.338		
CCcycle	-0.792	-0.853	0.900	0.657	
CCdensity	0.162	0.115	-0.724	-0.066	-0.463

Table 10 F-ratio matrix of all quantitative measures given by all analysis of variance presented above (the 3 columns are the SOURCE of variance).

	"experience"	"tasks"	"exp.x tasks"
task solving time	45.283	4.013	1.481
# of dialog states	8.830	5.789	0.831
# of transitions	6.682	2.869	0.390
CCstate	6.757	5.449	1.224
CCfan	29.026	1.667	0.423
CCcycle	21.547	1.405	0.285
CCdensity	4.355	31.621	1.438

We introduce the idea of "discriminating power" of the measures above. This discriminating power can be expressed as an F-ratio. Our definition of "discriminating power" is positively correlated with the F-ratio value given in an analysis of variance. We presume that the measure CC is a more or less stable attribute of the user in the scope of our study. CC is changing only during a learning process. We do not assume that the experts in our investigation acquire fresh knowledge of operating the interactive system during the task solving period. This assumption is valid, because the experts are highly skilled over several years of operating the database system ADIMENS.

To compare the discriminating power of both of the remaining measures CC_{fan} and CC_{cycle} we give an overview of the F-ratio of the three sources of variance given by the analysis of variance (see Tab. 10). Overall, the measures CC_{fan} and CC_{cycle} discriminate sufficiently between beginners and experts and not between the four tasks.

If we take into account that the measure TC_{fan} is not really appropriate to differentiate between the tasks, we can accept that the measure CC_{cycle} meets all demands. Overall, the best quantitative measure of complexity in our context seems to be CC_{cycle}. This measure estimates the average number of basic cycles in a net structure. Most empirical values of this measure lie between 195 and 215 near to the maximum of 215, which indicates that only some expert users had a complete knowledge of the system structure (see Fig. 5). To reach the maximum value of SC is only possible, when BC is equal to TC.

We can conclude that the four different quantitative measures of complexity we tested are of different value in measuring task and cognitive complexity. The measure of McCabe (1976) seems to be the most appropriate measure.

Acknowledgments

We gratefully acknowledge the fruitful discussions with Gudela Grote, the valuable remarks of Andrew Shepherd and the great support in developing the different programs by Thomas Greutmann, Jens Hofmann, Jack Rudnik and Martin Roth.
The preparation of this paper was supported by the BMFT (AuT programme) grant number 01 HK 706-0 as part of the BOSS "User oriented Software Development and Interface Design" research project.

References

ADIMENS (1988). A relational database system Version 2.21d. ADI Software GmbH, Hardeckstrasse 5, D-7500 Karlsruhe, Germany.

Benyon D. (1992). The role of task analysis in systems design. Interacting with Computers, 4(1):102-123.

Card S.K., Moran T.P. & Newell A. (1983). The psychology of human computer interaction. Erlbaum.

Crellin J , Horn T., Preece J. (1990). Evaluating Evaluation: A Case Study of the Use of Novel and Conventional Evaluation Techniques in a Small Company. Human-Computer Interaction - INTERACT '90 (Diaper D et al.; eds.) Elsevier, 329-335.

Dähler J. (1989). PACE user's manual. Pace Inc., Neptunstrasse 16, CH-8280 Kreuzlingen, Switzerland.

Genrich H.J., Lautenbach K. & Thiagarajan P.S. (1980). Elements of general net theory. Lecture Notes in Computer Science vol. 84 "Net Theory and Applications" (Bauer, W.; ed.) Springer, 21-163.

Greif S. & Gediga G. (1987). A critique and empirical investigation of the "one-best-way-models" in human-computer interaction. Psychological Issues of Human-Computer Interaction in the Work Place (Frese M., Ulich E. & Dzida W.; eds.) Elsevier, 357-377.

Grote F.G. & James L.R. (1990). A policy capturing approach to the measurement of cognitive complexity. European Perspectives in Psychology vol. I (Drenth P.J.D., Sergeant J.A. & Takens R.J.; eds.) Wiley & Sons, 417-434.

Hofmann J. & Rudnik J. (1991). PACEGEN: ein automatisches Logfile-Auswertungsprogramm zur Generierung von Petri Netzen. unpublished technical report, Work and Organizational Psychology Unit, Swiss Federal Institute of Technology (ETH).

Karat J. & Bennett J. (1991). Modelling the user interaction methods imposed by designs. Mental Models and Human-Computer Interaction (Tauber M.J. & Ackermann D.; eds.) Elsevier, 257-269.

Kieras D.E. & Polson P.G. (1985). An approach to the formal analysis of user complexity. International Journal of Man-Machine Studies, 22:365-394.

Kornwachs K. (1987). A quantitative measure for the complexity of man-machine interaction process. Proceedings of Human-Computer Interaction - INTERACT'87 (Bullinger H.-J. & Shackel B.; eds.) Elsevier (North-Holland). 109-116.

McCabe T. (1976). A complexity measure. IEEE Transactions on Software Engineering, SE-2(6):308-320.

McDaniel E. & Lawrence C. (1990). Levels of cognitive complexity: an approach to the measurement of thinking. Springer.

Moran T.P. (1981). The command language grammar: a representation for the user interface of interactive computer systems. International Journal of Man-Machine Studies, 15:3-50.

Payne S.J. & Green T.G.R. (1986). Task-action grammars: a model of the mental representation of task languages. Human Computer Interaction, 2:93-133.

Pervin L.A. (1984). Personality. Wiley.

Petri C.A. (1980). Introduction to general net theory. Lecture Notes in Computer Science vol. 84 "Net Theory and Applications" (Bauer, W.; ed.) Springer, 1-19.

Reisner P. (1981). Formal grammar and human factors design of an interactive graphics system. IEEE Transactions on Software Engineering, SE-7(2):229-240.

Reisner P. (1984). Formal grammar as a tool for analyzing ease of use. Human Factors in Computing Systems (Thomas J.C. & Schneider M.L.; eds.) Ablex, 53-78.

Scott W.A., Osgood D.W. & Peterson C. (1979). Cognitive structure: theory and measurement of individual differences. Wiley.

Stevens W.P., Myers G.J. and Constantine L.L. (1974). Structured design IBM System Journal, 13(2):115-139.

Sutcliffe A. (1989). Task analysis, systems analysis and design: symbiosis or synthesis? Interactng with Computers, 1(1):6-12.

Ulich E., Rauterberg M., Moll T., Greutmann T. & Strohm O. (1991). Task orientation and User-Oriented Dialog Design. International Journal of Human-Computer Interaction, 3(2):117-144.

Wasserman A.I. (1985). Extending state transition diagrams for the specification of human-computer interaction. IEEE Transactions on Software Engineering, SE-11(8):699-713.