Learning in Man-Machine Systems:
the Measurement of Behavioural and Cognitive Complexity

Matthias Rauterberg and Roger Aeppli

Work and Organisational Psychology Unit
Swiss Federal Institute of Technology (ETH)
Nelkenstrasse 11, CH-8092 ZURICH
Switzerland

ABSTRACT

A learning experiment was carried out to investigated the development of mental structures. Six subjects were carefully instructed to operate a commercial database management system (DBMS). The long-term knowledge about general DBMS- and computer experience was measured with a questionnaire once at the beginning of the investigation. On three weeks in a row all subjects had to solve the same task twice repeated in an individual session, overall there are six solutions of the same task. At the beginning of each of the three individual sessions the short-term knowledge about the task and the tool was measured with a short questionnaire. For each task solving process all keystrokes were recorded with a time stamp in a logfile. With a special analysing program the logical structure of each empirically observed task solving process was extracted. This logical structure is given as a Petri net. The behavioural complexity (BC) of this net structure can be measured with the McCabe-measure. With some special assumptions the cognitive complexity (CC) can be derived from the empirically gained BC. The main results are: (1) The time structure and BC measure different aspects of the learning process; (2) the time structure is–overall–positively correlated with BC and negatively correlated with CC; and (3) as well the long-term- as the short-term knowledge has an increasing predictive power with the time structure, but not with BC and CC.

1. INTRODUCTION

We live in a dynamic and irreversible changing world. We are information processing systems and have a huge learning potential. We have to realise and to accept that humans do not stop learning after end of school. We are compelled to learn and to make experiences our whole life. In his law of requisite variety Asby [1] pointed out, that for a given state of the environment, an open system has to be able to respond adaptively, otherwise the adaptability and the ability of the system to survive is reduced. A learning system, without input or with constant input, either decays or (in the best case) remains the same. Learning and the need for variety implies, that with constant variety of the context the requisite variety of the system tends to decay over time. In man-computer interaction we are able to measure the variety and complexity of human behaviour (e.g., explorative activities). With some plausible assumptions we are also able to estimate the complexity of users' mental model [11]. The complexity of the context (the internal structure of the interactive tool, e.g. software) can be measured, too.

2. LEARNING AND ACTIVITY

Learning increases constantly the complexity of the mental model. This is an irreversible process. One consequence is, that the contextual complexity must increase appropriately to fit the human needs for optimal variety. Based on the empirical result in [12], that the complexity of the observable behaviour of novices is larger than the complexity of experts, we concluded that the behavioural complexity is negatively correlated with the complexity of the mental model. Thus it is possible to estimate the cognitive complexity based on the measurement of the behavioural complexity, the measurement of the system complexity and the measurement of the task complexity (for a more detailed discussion see [11]).

3. THE MEASUREMENT OF COMPLEXITY

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 'system states'). The mental model of the user can be structured in representing: objects, operations, states, system structure, decision and task structure. Bauman and Turano [2] showed, that Petri nets are equivalent to formalism based on production rules (like CCT of Kieras and Polson [7]). In this sense, our approach can be subsumed under 'logic modelling', too.

A Petri-net can be described as a mathematical structure consisting of two non-empty disjoint sets of nodes (S-elements and T-elements), and a binary flow relation (F). The flow relation links only different node types and leaves no node isolated [10]. Petri nets can be interpreted in our context by using a suitable pair of concepts for the sets S (signified by a circle '( )') and T (signified by a square '[ ]') and a suitable interpretation for the flow relation F (signified by an arrow '->'). The main operations (relations) between two Petri nets are abstraction, embedding and folding [5].

The *folding operation* in the Petri-net theory is the basic idea of our approach. 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. Each state corresponds to a system context, and each transition corresponds to a system operation. This sequence is called a 'process' (see Figure 1). An *elementary process* is the shortest meaningful part of a sequence: (s') -> [t'] -> (s"). If the observable behaviour can be recorded in a complete ...-> (state) -> [transition] -> (state) ->... process description (see Figure 1), then the analysis and construction of the net structure of this process are simple: you have only to count the number of all different states and transitions used, or to mark on a list the frequencies of each state and transition used in the process. But, if the observable behaviour can only be recorded in an incomplete (e.g., ...-> (state) -> [transition] -> [transition] ->... or ...-> (state) -> (state) -> [transition] ->...) process description, then the analysis and construction of the net structure of this process are difficulty. You have to find out the correct state (transitions, resp.) between both transitions (states, resp.). Unfortunately, this is the most frequent case in practice. For these cases we need automatic tool support.

The aim of the 'folding' operation is to reduce the elements of an observed empirical interaction process to the minimum number of states and transitions, with the reduced number of elements being the 'logical decision structure'. Folding an interactive process ex-

tracts the embedded net structure and neglects the information of the amount of repetition and of the sequential order ('time structure').
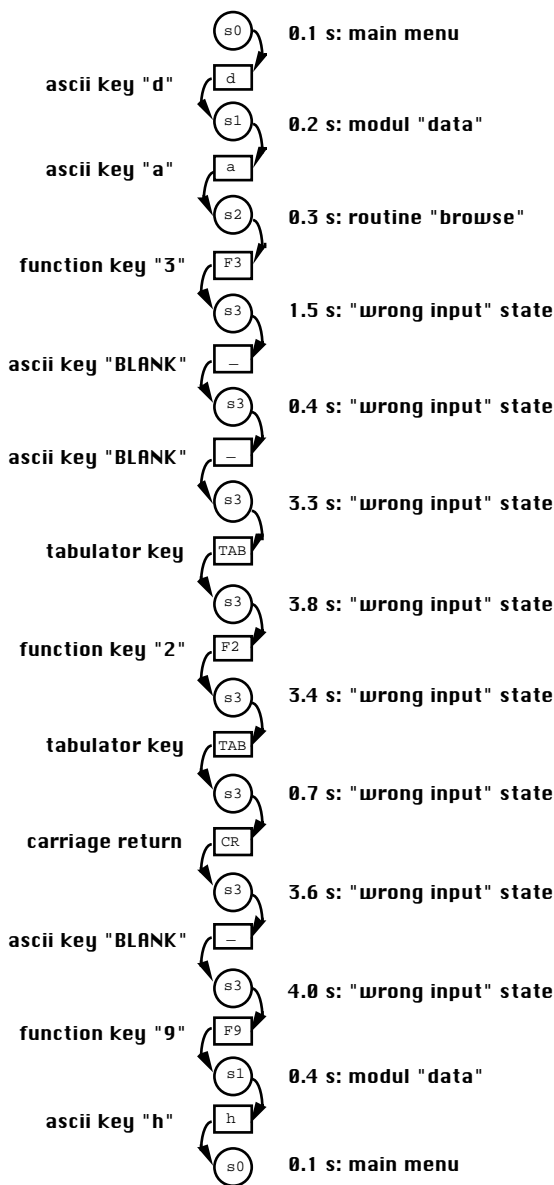

	s0	0.1 s: main menu
ascii key "d"	d	
	s1	0.2 s: modul "data"
ascii key "a"	a	
	s2	0.3 s: routine "browse"
function key "3"	F3	
	s3	1.5 s: "wrong input" state
ascii key "BLANK"	_	
	s3	0.4 s: "wrong input" state
ascii key "BLANK"	_	
	s3	3.3 s: "wrong input" state
tabulator key	TAB	
	s3	3.8 s: "wrong input" state
function key "2"	F2	
	s3	3.4 s: "wrong input" state
tabulator key	TAB	
	s3	0.7 s: "wrong input" state
carriage return	CR	
	s3	3.6 s: "wrong input" state
ascii key "BLANK"	_	
	s3	4.0 s: "wrong input" state
function key "9"	F9	
	s1	0.4 s: modul "data"
ascii key "h"	h	
	s0	0.1 s: main menu

Figure 1. Part of the interactive behaviour of an expert with a relational database system. The whole process of this example is based on 12 transitions and 12+1=13 states. The number on the right side is the 'time per key' in seconds.

A simple pattern matching algorithm looks for all 'elementary processes' in the sequence (cf. [12]). A composition algorithm (the folding operation) is now able to build up the Petri net combining all elementary processes. The result of a folding operation of our example sequence (see Figure 1) is the Petri net given in Figure 2. This special net with four different states and nine different transitions is the minimal net structure to reproduce the process given in Figure 1 (for a more detailed discussion see [14]). Measurable features of the behavioural process are: number of states and transitions totally used, number of different states and different transi-

tions used, planning time per state and transition, etc. These measurements can be easily done based on a protocol of the user's behaviour automatically recorded by an interactive software program (the dialog system) in a 'logfile'.

To measure complexity we use the C_{cycle} metrics of McCabe [8]. With C_{cycle} we have a useful quantitative metric to measure complexity. We are discussing the advantages and disadvantages of four different quantitative metrics in the context of an empirical investigation elsewhere (see [11]). The complexity measured with C_{cycle} 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 1 in the case of a sequence (T − S = −1); the value of P in our context is 1.

$$C_{cycle} = T - S + P \qquad \text{with } T \geq (S\text{-}1) \qquad (1)$$

The measure C_{cycle} of the example in Figure 2 is six [9 − 4 + 1 = 6]. But, what could this number mean? McCabe [8] interprets C_{cycle} as the *number of linear independent paths* through the net. Other interpretations of C_{cycle} are *number of holes* in a net or *number of alternative decisions* carried out by the users.

Observing the interactive 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.
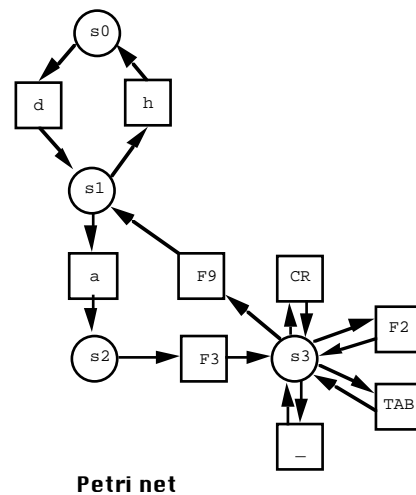

Petri net

Figure 2. The 'folded' Petri net of our example in Figure 1.

We call the complexity of the observable behaviour the 'behavioural complexity (BC)'. This behavioural 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

4686

(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 means, that the system structure constrains the complexity of the observable task solving space. Now we can state with the constraints (BC≥TC) and (SC≥CC), that:

$$BC - TC = SC - CC \qquad (2)$$

To get CC we transform this Formula 2 in Formula 3:

$$CC = SC + TC - BC \qquad (3)$$

The parameters SC and TC can be estimated either in a theoretical or in an empirical way. The parameter SC is given by the concrete system structure, so we have to apply a given complexity measure to this system structure (theoretical way). The empirical way to estimate SC is to take the *maximum* of all observed BCs per task. To calculate TC, all a priori descriptions of task structures are usable (theoretical way). If we have empirical data of different task solving processes of different persons, then we can estimate TC using the *minimum* per task of all observed BCs. 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.

4. A LEARNING EXPERIMENT

Subjects
Six users (6 men; average age of 25 ± 3 years) had to solve one task in two different versions on three weeks in a row with exactly seven days between each of the three task solving sessions.

Tasks
The dialog system of a commercial database management system (DBMS) with a character-oriented user-interface (CUI) and a hierarchical menu structure was the test system (for a detailed description see [13]). To solve the task, a list must be generated with four different data fields that have to be interactively selected. One data field must be calculated with a set of predefined calculation instructions stored in a disk-file. This list must be outputted to the screen and to the printer. The difference between task-1 and task-1' was only on the formulation level of the instruction, but on the structural level all six tasks were identically.

Procedure
The duration of the actual task solving session was about 15 minutes. At the first session each user had to fill out a questionnaire about his/her general computer experiences and his/her specific experiences with DBMS's and CUI's. Each user was carefully instructed to operate the DBMS, especially to solve the test task (23 ± 4 min of instruction time). At the beginning of each task solving session a short questionnaire about the correct task solving process must be answered. During the task solving session each keystroke with a time stamp was recorded in a logfile. Each user needed 68 ± 12 minutes for the whole experiment (each session was carried out as follows: two similar tasks, non speed condition, individual sessions, session no. = week no.).

Measures
We measured three different aspects: (1) once the general experience with computer and DBMS (the long-term knowledge), (2)

three times the actual knowledge about the task and the tool (the short-term knowledge per week), and (3) six times the task solving time and BC (performance measures per task).

Long-term knowledge: The questionnaire to measure the general computer and DBMS experience consists of twelve single questions. For each question the user has to estimate the number of hours spending on the asked topic in his/her life.

Short-term knowledge: The questionnaire to measure the actual knowledge about the task-tool mapping consists of three lists with 24 DBMS-functions. The user has to solve three sub-tasks by numbering all appropriate functions in the correct sequential order. The correspondence of the answered sequence with the correct solution was calculated with a *similarity ratio* (range 0…1; see [14]).

Complexity: Based on the logfiles we could measure BC with our analysing tool AMME (cf. [12]). To estimate CC (see Formula 3) we have to determine SC and TC. SC can be estimated as the maximum of all observed BC's (e.g., SC = 28). TC can be estimated as the minimum of all observed BC's (e.g., TC = 22; see Figure 4).

Results of the analysis of variances
Long-term knowledge: On average 2698 ± 2534 hrs of general experience with computer technology was measured (including 274 ± 601 hrs of DBMS experience, and 1693 ± 2112 hrs CUI experience). The product moment correlation between DBMS experience and general experience is $r = .43$ ($p \leq 0.42$, N=6), and between CUI experience and general experience is $r = .85$ ($p \leq 0.03$, N=6).

Short-term knowledge: The similarity ratio as a measure of the actual knowledge about the task-tool mapping ('task tool knowledge') was neither significantly different between the three weeks (mean$_{week-1}$ = 0.89 ± 0.13, mean$_{week-2}$ = 0.84 ± 0.16, mean$_{week-3}$ = 0.82 ± 0.22, $F_{(2,15)} = 0.39$, $p \leq .685$), nor between the three sub-tasks (mean$_{subtask-A}$ = 0.85 ± 0.16, mean$_{subtask-B}$ = 0.80 ± 0.18, mean$_{subtask-C}$ = 0.90 ± 0.17, $F_{(1,30)} = 2.40$, $p \leq .108$). No significant interaction term exists ($F_{(4,30)} = 0.35$, $p \leq .840$).

Task solving time: The difference of the task solving time between task-1 and task-1' is significant (mean$_{task-1}$ = 9.5 ± 3.8 min, mean$_{task-1'}$ = 5.2 ± 0.9 min, $F_{(1,10)} = 15.3$, $p \leq .003$, see Figure 3).
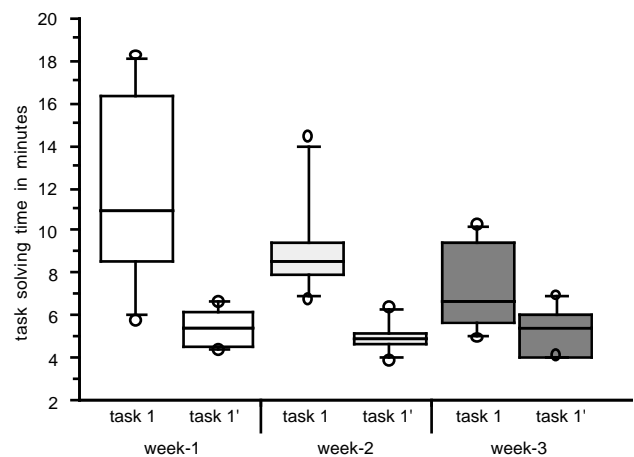

Figure 3. The Box plot of the average task solving time.

There is also a significant difference of the task solving time between the three weeks (mean$_{week-1}$ = 8.6 ± 4.7 min, mean$_{week-2}$ = 7.2 ± 2.9 min, mean$_{week-3}$ = 6.3 ± 2.0 min, $F_{(2,20)} = 3.6$, $p \leq .045$).

4687

The interaction term is nearly significant ($F(2,20) = 3.2$, $p \le .061$; see Figure 3). The decline of the task solving time for task-1 over the three weeks corresponds to the well-known learning curve (see [6]). But the nearly significant interaction term can be interpreted as if all users learned the correct solution already for task-1' in the first session.

Behavioural complexity BC: The difference of BC between task-1 and task-1' is significant ($\text{mean}_{\text{task-1}}= 24.3 \pm 1.7$, $\text{mean}_{\text{task-1'}}= 22.6 \pm 1.4$, $F(1,10) = 12.7$, $p \le .005$; see Figure 4). All other differences are not significant (see Figure 4). Except one, all users had already a BC of 23 in task-1' of the first session. This result corresponds with the result of the task solving time. All users reached their minimum of BC = 22 in task-1' of the second session. It is important to note that from that moment the variance disappears.
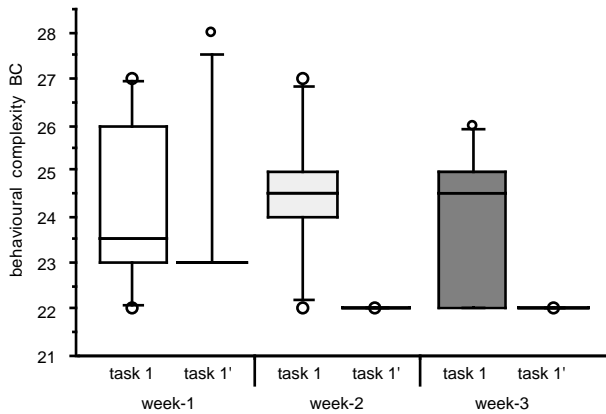
Figure 4. The Box plot of the average behavioural complexity BC.

Cognitive complexity CC: CC is exactly with $r = -1.0$ negatively correlated with BC. Therefore, only the difference of CC between task-1 and task-1' is significant, too ($\text{mean}_{\text{task-1}}= 25.7 \pm 1.7$, $\text{mean}_{\text{task-1'}}= 27.4 \pm 1.4$, $F(1,10) = 12.7$, $p \le .005$). All other differences are not significant (see Figure 5). All users reached their maximum of CC = 28 in task-1' of the second week (see Figure 5).
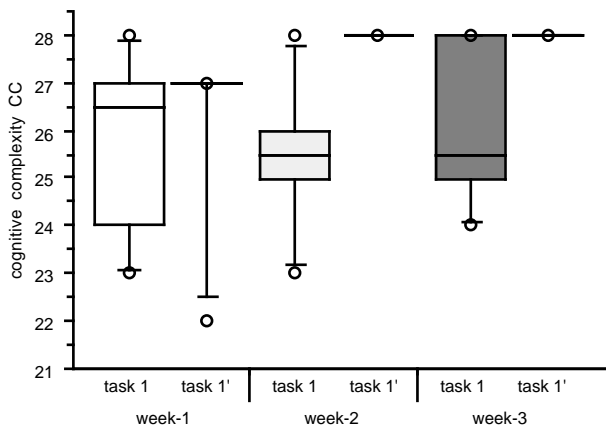

Figure 5. The Box plot of the average cognitive complexity CC.

Results of the correlation analysis
To estimate the influence of the general experience on the learning process, and to observe the development process of the actual task-

tool mapping all relevant product moment correlations are calculated and presented in Figure 6 to Figure 9.

In the first session ('1. week') the normally positive correlation between $\text{time}_{\text{task-1}}$ and $\text{BC}_{\text{task-1}}$ ($r = +.69$ in Figure 6; $r = \pm .87$ in Figure 7, and $r = +.80$ in Figure 8) changes to a negative correlation between $\text{time}_{\text{task-1}}$ and $\text{BC}_{\text{task-1'}}$ ($r = -.48$, see Figure 6). This result is caused by one user with the most DBMS experience who produced the most complex net (BC=28) in the shortest time (4.5 min).
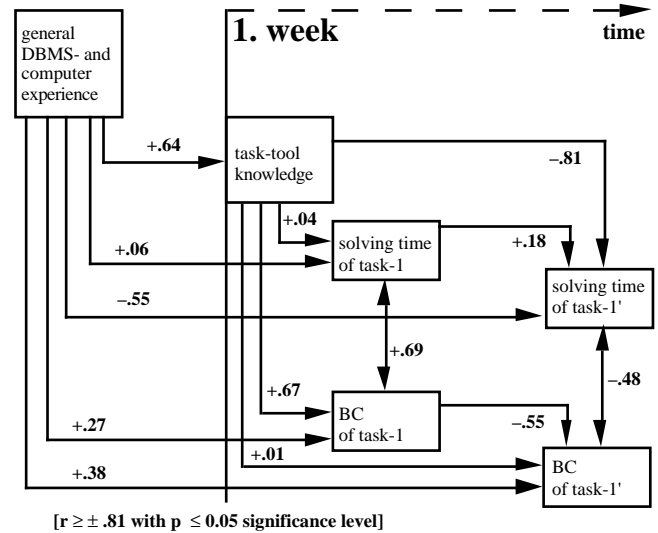

[$r \ge \pm .81$ with $p \le 0.05$ significance level]

Figure 6. Diagram with all relevant product moment correlation's r between long-term (general experience) and short-term knowledge (task-tool) and both measures 'time' and 'BC' of the first week.
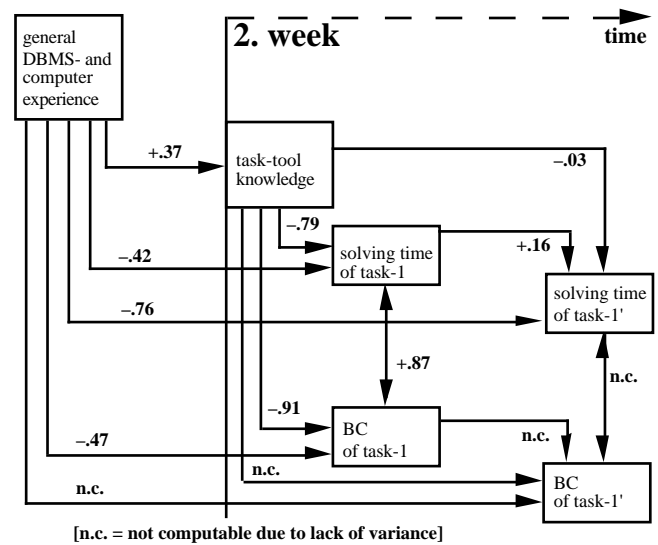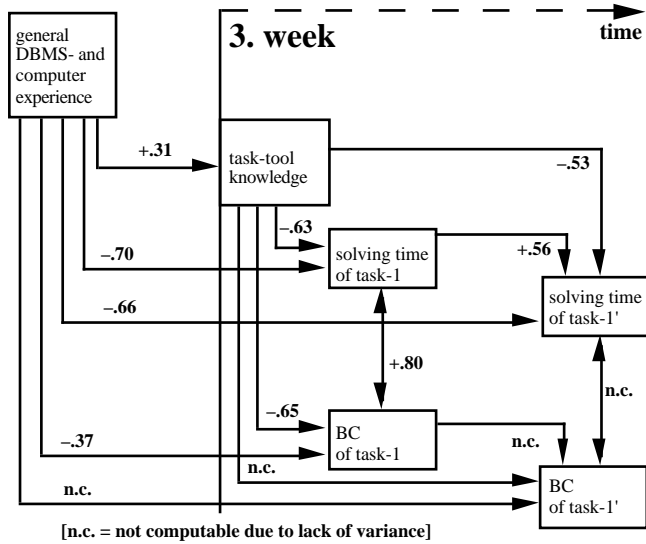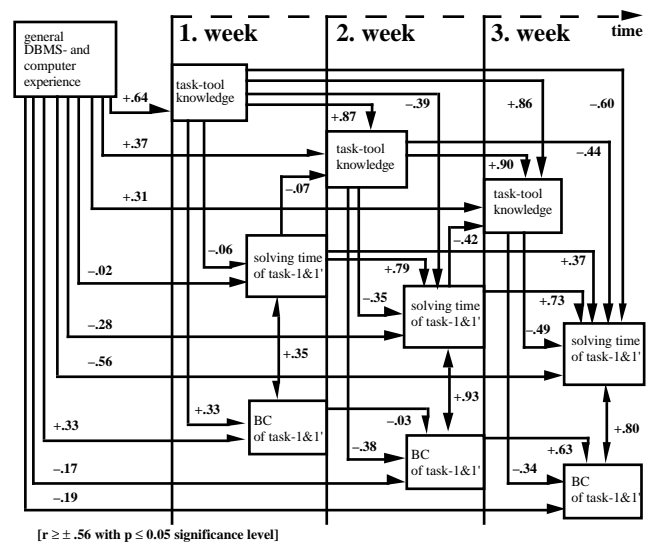

[n.c. = not computable due to lack of variance]

Figure 7. Diagram with all relevant product moment correlation's r between long-term (general experience) and short-term knowledge (task-tool) and both measures 'time' and 'BC' of the second week.

If we compare the correlation pattern of week-1 (cf. Figure 6) with the correlation pattern of week-2 (cf. Figure 7) and week-3 (cf. Figure 8) then we can observe a qualitative change from a positive to

a negative correlation of the long-term and of the short-term knowledge with BC (see also Figure 9). We interpret this result as exploratory behaviour during the first session: the more knowledge, the greater the tendency to explore. This interpretation is congruent with the result of the most experienced DBMS user discussed in the last paragraph. Due to lack of variance for week-2 and week-3 the correlation between $BC_{task-1'}$ and the other measures can not be computed.

[n.c. = not computable due to lack of variance]

Figure 8. Diagram with all relevant product moment correlation's r between long-term (general experience) and short-term knowledge (task-tool) and both measures 'time' and 'BC' of the third week.

[r ≥ ± .56 with p ≤ 0.05 significance level]

Figure 9. Diagram with all relevant product moment correlation's r between long-term (general experience) and short-term knowledge (task-tool) and both measures 'time' and 'BC' for all three weeks.

To get an overview over the development of the different influences of long- and short-term knowledge on task solving time and BC, we calculated all relevant correlations (collapsing task-1 and task-1'; see Figure 9). The influence of long-term knowledge on task time is increasing from week to week ($r = -.02$ week$_{1x1}$, $-.28$ week$_{1x2}$, and $-.56$ week$_{1x3}$; see Figure 9).

A very similar effect can be observed for the influence of the short-term knowledge about the task-tool mapping on task solving time ($r = -.06$ week$_{1x1}$, $-.35$ week$_{2x2}$, and $-.49$ week$_{3x3}$; see Figure 9). The measurement of the short-term knowledge about the task-tool mapping shows a high retest reliability (inter-test correlation $r = +.87$ week$_{1x2}$, $+.86$ week$_{1x3}$, $+.90$ week$_{2x3}$, see Figure 9).

Both–long-term and short-term knowledge–change from a positive to a negative correlation from week-1 to week-2 and week-3. If we assume that BC is a valid measure for the amount of exploratory behaviour, then we can come up with the following plausible interpretation for this result: First, an exploratory behaviour helps the user to learn as much as possible about the system; during this first phase the long- and short-term knowledge enables and guides the exploration. After this first exploration phase the user can produce the minimal solution based on his new knowledge about the system. This interpretation could explain, why the correlation between BC of week-1 and BC of week-2 is zero and increases to a significant correlation between week-2 and week-3 ($r = -.03$ week$_{1x2}$, $+.63$ week$_{2x3}$, see Figure 9).

5. DISCUSSION

The three main results are: (1) The time structure and BC measure different aspects of the learning process; (2) the time structure is–overall–positively correlated with BC and negatively correlated with CC; and (3) as well the long-term knowledge as the short-term knowledge has an increasing predictive power with the time structure, but not with BC and CC.

First, over the three weeks the time structure of task-1 shows the well-known learning curve ([6], see Figure 3). On the other side, the time structure of task-1' reaches its minimum already in the first session. The logical structure–measured with BC–of task-1 varies around a median of 24 over the three weeks (see Figure 4). Only BC of task-1' shows a remarkable decrease in the first session (BC=23), and reaches its minimum in the second session (BC=22; cf. Figure 4). Our results show that the development of the time structure is different from the development of the logical task structure (cf. Figure 3 and Figure 4). After the first learning trial we can observe a variation in task solving time of the task-1', but not in its complexity. This effect should be investigated further on to test the assumption "that it is rules that change with practice, not the speed of execution of the rules" [3].

Second, at a first glance a positive correlation between task time and BC sounds trivial: of course, the greater BC is, the more task solving time is necessary (see Figure 9). But, if we assume that our assumption to estimate CC is correct, then this positive correlation between task time and BC leads directly to a *negative* correlation of the same size between task solving time and CC! The more a user knows about the system or context, the less time he/she needs to interact with the system. Learning to behave in a complex environment means to increase the efficiency and to decrease the complexity of the interaction process with the context (cf. Figure 10). This result is in contradiction to the following assumption in the community that works on modelling of user behaviour: The "mental model maps completely to the relevant part of the conceptual model, e.g. the user virtual machine. Unexpected effects and errors point to inconsistency between the mental model and the conceptual model" ([16] p. 258). And a second voice: "… the central assumption … is that the user's mental representation corresponds to that given in instruction" ([4], p. 4). Of course, an adequate mental

4689

model contains all task and tool relevant knowledge, but something in addition. Cognitive complexity is–overall–negatively correlated to behavioural complexity and task efficiency. What is the additional knowledge of? Why is this knowledge not observable in the concrete task solving process? We assume–and there is some empirical evidence (see [15])–that these additional parts of our memory consist of knowledge about unsuccessful behaviour (e.g. faults, errors, etc.). One advantage of our approach is that we do not differentiate between errors and correct task solving behaviour. All kinds of observable actions (e.g. errors and mistakes) are included in our analysis and measured with BC.
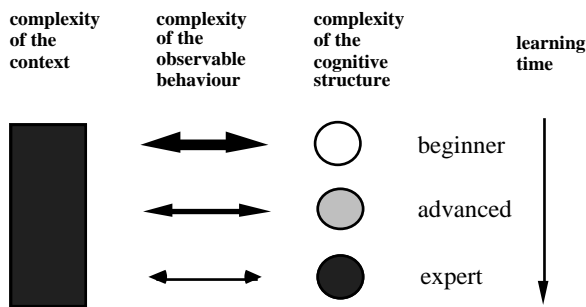
Figure 10. A schematic overview about knowledge acquisition and 'behavioural complexity'.

Third, in the first learning trial (task-1) of the first session the *positive* correlation between long- and short-term knowledge on one side and BC on the other side can be interpreted as an exploratory phase guide by prior experience (see [9]). After this exploratory phase the behaviour changes to an optimisation activity: looking for an optimal task solving procedure. In this sense we can understand the increasing predictive power of long- and short-term knowledge on task solving time of week-2 and week-3.

6. CONCLUSION

Three important conclusions can be drawn: (1) Our assumption–that 'learning how to solve a specific task with a given system means that behavioural complexity (BC) decreases and cognitive complexity (CC) increases'–seems to be correct. (2) After an exploratory phase all users reached the correct and minimal task solution (minimum of BC = 22, see Figure 4), and the variance between users disappeared. Furthermore, (3) we can conclude from the empirical results that we must discriminate between the *time structure* [6] and the *logical structure* [14] of a task. The time structure can be measured with the task solving time. The logical structure of a task can be extracted with the special analysing tool AMME [12], and the complexity of this logical structure can be measured with the Mc-Cabe-measure (BC) [8]. At week-2 all users learned completely the logical task structure (minimum of BC, no variance, see Figure 4) although the task solving time decreases further on. Learning the time structure means to accelerate the task solving process. One psychological interpretation of the acceleration of the time structure seems to be–if the correct task solving sequence was learned–that self-confidence grows and comes strong in doing the right things at the right time.

REFERENCES

[1] R.W. Ashby, "Requisite variety and its implications for the control of complex systems", *Cybernetica*, Vol. 1, No. 2, 1958, pp. 1-17.

[2] R. Bauman and T.A. Turano, "Production based language simulation of Petri nets", *Simulation*, Vol. 47, 1986, pp. 191-198.

[3] S. Bovair, D. Kieras and P.G. Polson, "The acquisition and performance of text-editing skill: a cognitive complexity analysis", *Human-Computer Interaction,* Vol. 5, 1990, pp. 1-48.

[4] E. Churchill, "The formation of mental models: are 'device instructions' the source?", in: G.C. van der Veer, M.J. Tauber, S. Bagnara and A. Antalovits, Eds., *Human-Computer Interaction: Tasks and Organisation* (CUD, Roma, 1992, pp. 3-16).

[5] H.J. Genrich, K. Lautenbach and P. S. Thiagarajan, "Elements of general net theory", *Lecture Notes in Computer Science*, Vol. 84, 1980, pp. 21-163.

[6] S.L. Johnson, "Using mathematical models of the learning curve in training system design", in: *Proceedings of the Human Factors Society 29th Annual Meeting*, Vol. II, 1985, pp. 735-739.

[7] D.E. Kieras and P.G. Polson, "An approach to the formal analysis of user complexity", *International Journal of Man-Machine Studies*, Vol. 22, 1985, pp. 365-394.

[8] T. McCabe, "A complexity measure", *IEEE Transactions on Software Engineering*, Vol. SE-2, 1976, pp. 308-320.

[9] H. van Oostendorp and B. J. Walbeehm, "Towards modelling exploratory learning in the context of direct manipulation interfaces", *Interacting with Computers*, Vol. 7, No. 1, 1995, pp. 3-24.

[10] C.A. Petri, "Introduction to general net theory", *Lecture Notes in Computer Science*, Vol. 84, 1980, pp. 1-19.

[11] M. Rauterberg, "A method of a quantitative measurement of cognitive complexity", in: G.C. van der Veer, M.J. Tauber, S. Bagnara and A. Antalovits, Eds., *Human-Computer Interaction: Tasks and Organisation* (CUD, Roma, 1992, pp. 295-307).

[12] M. Rauterberg, "AMME: an automatic mental model evaluation to analyze user behaviour traced in a finite, discrete state space", *Ergonomics,* Vol. 36, No. 11, 1993, pp. 1369-1380.

[13] M. Rauterberg, "An empirical comparison of menu-selection (CUI) and desktop (GUI) computer programs carried out by beginners and experts", *Behaviour and Information Technology*, Vol. 11, No. 4, 1992, pp. 227-236.

[14] M. Rauterberg, "From novice to expert decision behaviour: a qualitative modelling approach with Petri nets", in: *Proceedings of 6th International Conference on Human-Computer Interaction - HCI'95* in Yokohama (J), July 9-14, 1995 (in press).

[15] M. Rauterberg, "About faults, errors, and other dangerous things", in: *Proceedings of the Symposium on Human Interaction with Complex Systems* in Greensboro (USA), Sept. 17-20, 1995 (in press).

[16] G. Van der Veer, S. Guest, P. Haselager, P. Innocent, E. McDaid, L. Oesterreicher, M. Tauber, U. Vos & Y. Waern, "Desiging for the mental model: an interdisziplinary approach to the definition of a user interface for electronic mail systems", in: D. Ackermann and M. Tauber (Eds.) *Mental Models and Human-Computer Interaction 1* (North-Holland, Amsterdam, 1990, pp. 253-288).

1995 IEEE International Conference on Systems, Man and Cybernetics

Intelligent Systems for the 21st Century

Vancouver, British Columbia

Canada

October 22 25, 1995