M. Fjeld, S. Schluep & M. Rauterberg (1998): Automatic, action driven classification of problem solving strategies by statistical and analytical methods: a comparative study. In F. E. Ritter & R. M. Young (eds.) Proceedings of the Second European Conference on Cognitive Modelling, pp. 98-103. Nottingham: Nottingham University Press.

Automatic, action driven classification of user problem solving strategies by statistical and analytical methods: a comparative study

M. Fjeld, S. Schluep & M. Rauterberg
Institute of Hygiene and Applied Physiology (IHA)
Swiss Federal Institute of Technology (ETH)
Clausiusstr. 25, CH-8092 Zurich
{fjeld, schluep, rauterberg}@iha.bepr.ethz.ch

ABSTRACT

We have recorded the behaviour of several users solving the same tasks with an interactive database program and were able to identify several distinct strategies. Since the number of users exceeds the number of strategies, multiple users will have a strategy in common. Our aim was to find groups of users sharing the same strategy. Following each of the three methods (correlation, intersection, and exclusion) we define a metric among task solving sequences. For multiple users, we represent these measures by a matrix system, in order to find groups of users with common behaviour. Direct interpretation or multi dimensional scaling of such matrices indicates distinct user groups. The common denominator for each group can be interpreted as a strategy. A few distinctive solution strategies were found to exist.

Keywords

Mental models, observable behaviour, plan recognition, user strategies, statistical analysis, repetitive behaviour

1 MODELLING APPROACH

Humans express themselves in many ways. One of these ways is everyday problem solving. We will focus on problem solving in the domain of human computer interaction. In particular, we will examine how multiple users solve various tasks with a relational database application.

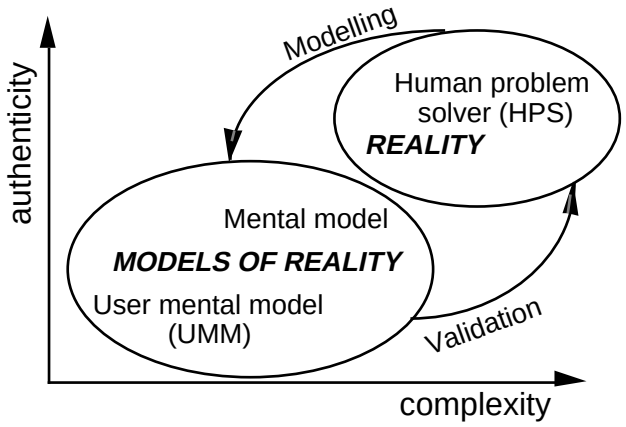

Fig. 1: A scheme showing the differences between models of reality and real humans (HPSs). Models are meant to represent objects and processes existing in reality.

It is hard to grasp how human problem solvers (HPS) really express themselves, since the persons we study are live beings. Nevertheless, a mental model (see Fig. 1) may give us an idea of the real HPS. Since we are interested in computer mediated, everyday task solving, we introduce a special case of mental models, called user mental model (Tauber, 1985) (UMM; see Fig. 1). UMMs can bring understanding about the strategies people use when solving specific problems. UMMs can be represented in many ways, using plain text, Petri nets or state-transition vectors. We choose the latter representation to elaborate UMMs based on observable task solving behaviour.

In general, we observe a lot of task solving behaviour that is not strictly *task related*. If we study one user solving a task, we are hardly able to single out the successful *strategy* from the *remaining behaviour*. One approach may be to study many users solving the same task. Since they all solve the same problem, we suppose that their common behaviour is what was required to solve the task. If there are several successful strategies, some users may have one strategy in common, other users a second one.

Successful strategies are most often defined by the given task-system combination. For users to accomplish a task, they must follow one of these strategies. As soon as a successful strategy has been accomplished, user behaviour is finished.

Which strategy a user prefers, as well as other kinds of user behaviour can tell us something about the particular HPS; for instance how the successful strategy was acquired. Given a behavioural task solving sequence, we want to separate the *strategy* (which is more related to the task-system combination) from the *remaining behaviour* (which is more related to the HPS). In the rest of this paper, *strategy* will mean one (of many), possibly error free, task solving behavioural sequences.

The aim of our work is to find which strategies are needed to solve a given task. We are looking for automatic methods to find these strategies. Under certain conditions, strategies may also be obtained by protocol analysis (Ericsson and Simon, 1984). Protocol analysis implies manual inspection of video and verbal utterances in addition to logfiles. With simple tasks, this work can be overcome. For more complex tasks, protocol analysis becomes cumbersome. Semi-automatic generation of

M. Fjeld, S. Schluep & M. Rauterberg (1998): Automatic, action driven classification of problem solving strategies by statistical and analytical methods: a comparative study. In F. E. Ritter & R. M. Young (eds.) Proceedings of the Second European Conference on Cognitive Modelling, pp. 98-103. Nottingham: Nottingham University Press.

process models was studied by Ritter and Larkin (1994). Motivated by their work, we wish to suggest further principles for automatic recognition of user strategies and plans.

In this paper, human perception and verbalisation will not be considered as part of the problem solving. Hence, purely based on observable task solving behaviour, we set out for automatic methods, applicable with simple as well as with complex tasks. We only consider protocol analysis as a mean to validate the automatic methods we elaborate.

2 SYSTEM DESCRIPTION

The system we study is a relational database program with 153 different dialogue states. The possible transitions of the system are represented by a state-transition vector space. A state-transition-vector (STV) summarises a subject's task solving behaviour for one task. It has length n, where n is the total number of transitions (n=978) for the complete database program. Each STV element tells how many times a certain transition was activated to solve the task.

Since the order of activated transitions is not contained in the STV, the order of user behaviour is only partly conserved. It is stored in an implicit form, given by the system dialogue structure and is embedded in the structure of the STV.

To reduce complexity, it is possible to replace each STV element >1, by 1. We call the result binary-state-transition-vector (B-STV). It tells us which transitions were activated, but nothing about repetition.

3 TASK DOMAIN

An empirical investigation was carried out to compare different types of expertise (Rauterberg, 1992). For the reconstruction of UMMs we used logfiles of six novice and six expert users, all solving the same task. The task was to find out how many data records there are in a given database consisting of three file. An example UMM of a task solving process, based on one of the experts, is presented in Rauterberg et al. (1997). In that example, 15 *different* transitions (number of positive STV elements) were activated to solve the task. However, since some of them were activated repeatedly, the *total* number of activated transitions (the sum of STV elements) is 25.

4 INTERPRETING BEHAVIOURAL SEQUENCES

Studying an STV of one user can tell us which system states the user passed by, which transitions that were triggered in those states and how many times that happened. Different users working with the same system are directly comparable, since their behavioural sequences only differ by the value of the vector elements.

5 BASIC QUESTIONS AND METHODOLOGY

First, we want to find out how the behavioural sequences of two users can be related. A classical method is that of correlation. An alternative is to look for analytical methods. The user STVs can be represented by ellipses as in Fig. 2. The area of an ellipse corresponds to the sum of the STV element values. Intersection area can be understood as symmetric similarity between two user STVs. Exclusion areas can be understood as the asymmetric difference between two user STVs.

Based on such considerations, we raise the following questions and suggest corresponding methods as answer:

1) What is the proximity between two behavioural sequences? Method suggested: *correlation.*

2) What do two behavioural sequences have in common (similarity)? Method suggested: *intersection* (Fig. 2).

3) What do two behavioural sequences not have in common (difference)? Method suggested: *exclusion* (Fig. 2).

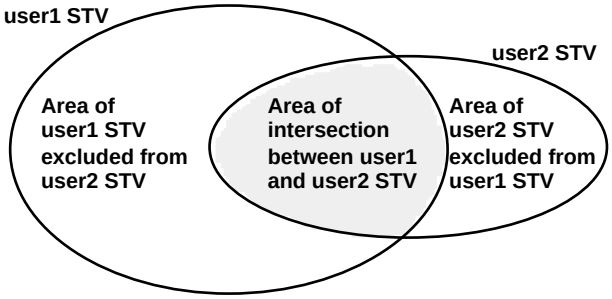

Fig. 2: Intersection area and exclusion areas between user1 and user2 STV.

For each method, we elaborate a metric (Table 1). The order of the metric may be symmetrical (the metric applied from user1 STV to user2 STV is the same as the metric applied from user2 STV to user1 STV) or asymmetric (the metric applied from user1 STV to user2 STV is *not* the same as the metric applied from user2 STV to user1 STV). Based on the metrics applied between all the user STVs, we then apply a grouping algorithm.

With each group suggested by the grouping algorithm, a strategy may be approximated. The procedure is to create a STV with a maximum number of non-zero elements common to all the users of the group.

In the following presentation, we will proceed from more statistically based to more analytically based methods.

Table 1: The three suggested methods and their characteristics. CORR means a standard correlation method, the other metrics are defined by Formula 1,2 and 3.

Method	Metric name	Metric nature	Grouping algorithm
Correlation	CORR	Statistical	Statistical
Intersection	$M^{IS}_{p,q}$, $M^{BIS}_{p,q}$	Analytical	Statistical
Exclusion	$M^{EX}_{p,q}$	Analytical	Analytical

5.1 CORRELATION METHOD

In this method the metric between user STVs is the degree of proximity. The metric values are analysed by multi-dimensional-scaling (MDS, Systat, 1989) to indicate groups of users.

5.1.1 METRIC

Correlation is one way to measure the proximity between behavioural sequences. We apply Pearson correlation as a measure for proximity between two STVs. By this procedure, we get an mxm (m=12) diagonal dominant symmetrical matrix with possible values between minus one, via zero (no proximity) and one (equality). For Fig. 3 the observed values are between -0.003 and 0.948 (without considering the diagonal elements).

5.1.2 GROUPING ALGORITHM

The correlation matrix is interpret by MDS, giving the plot of Fig. 3. We have chosen to apply two dimensional MDS to allow visual interpretation of the plots.

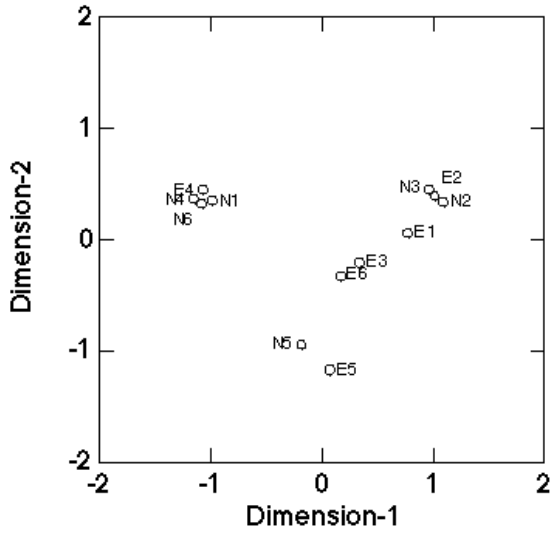

Fig. 3: MDS (r=1, Kruskal, Mono) plot with a Pearson correlation matrix gives RSQ=0.870.

5.1.3 OUTCOME

From the plot in Fig. 3 we see how the users may be grouped: {N1, N4, N6, E4}, {N2, N3, E1, E2, E3, E6} and {N5, E5}. Some of these user STVs may well consists of parts of several strategies in addition to the successful one.

According to the proportion of variance (RSQ=0.870), MDS explains some of the variance of the user data, but a significant part remains unexplained.

5.2 INTERSECTION METHOD

This method is based on the observation that if two users followed the same strategy, that strategy will belong to the intersection of the two users STVs. The order of the an intersection metric is symmetric, since both user STVs have the same in common. These metric values are analysed by MDS to indicate groups of users.

5.2.1 METRIC

Similar behaviour is measured by summing up the smaller STV elements of the two user STVs, thus considering the number of activated transitions common to both users.

It is reasonable to *normalise* the degree of intersection by the smaller of the sums of the STVs elements (which would be the maximum possible value for the intersection).

Formula 1:

$$M_{p,q}^{IS} = \frac{\sum_{i=1}^{n} \min\left(e_{p,i}, e_{q,i}\right)}{\min\left(\sum_{i=1}^{n} e_{p,i}, \sum_{i=1}^{n} e_{p,i}\right)}$$

where:

$M_{p,q}^{IS}$	: Intersection metric between user p and q
i	: Summing Index STV elements
n	: STV length, upper summing limit
$e_{p,i}$	: STV element i for user p
$e_{q,i}$	: STV element i for user q

We may ignore repetitive behaviour, using B-STVs instead of STV. Results based on B-STVs are called *binary*.

Formula 2:

$$M_{p,q}^{BIS} = \frac{\sum_{i=1}^{n} \min\left(e_{p,i} \cdot e_{q,i}, 1\right)}{\min\left(\sum_{i=1}^{n} \min\left(e_{p,i}, 1\right), \sum_{i=1}^{n} \min\left(e_{p,i}, 1\right)\right)}$$

where:

$M_{p,q}^{BIS}$	: Binary intersection metric between user p and q
i	: Summing Index B-STV elements
n	: B-STV length, upper summing limit
$e_{p,i}$	: B-STV element i for user p
$e_{q,i}$	: B-STV element i for user q

By this procedure, we get an mxm (m=12) symmetrical matrix with elements based on STVs (Formula 1) or B-STVs (Formula 2). The elements take possible values between zero (no similarity) and one (equality). For Fig. 4, based on STVs, the observed values are between 0.078 and 0.929 (without considering the diagonal elements). For Fig. 5, based on B-STVs, the observed values are between 0.182 and 0.882 (without considering the diagonal elements).

5.2.2 GROUPING ALGORITHM

We interpret the symmetrical exclusion matrix by MDS, obtaining plots like Figs. 4 and 5. The users seem to represent three groups, {N1, N4, N5, N6, E4}, {N2, N3, E1, E2, E5} and {E3, E6}. E3 and E6 may as well be combinations of several strategies.

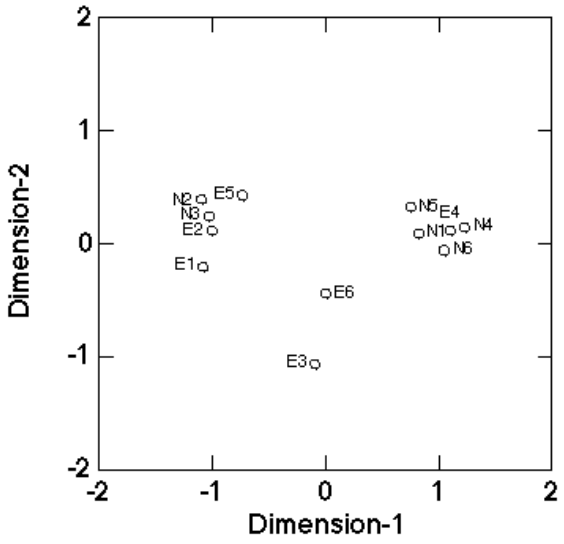

Fig. 4: MDS (r=1, Kruskal, Mono) plot with a normalised intersection matrix gives RSQ=0.975.

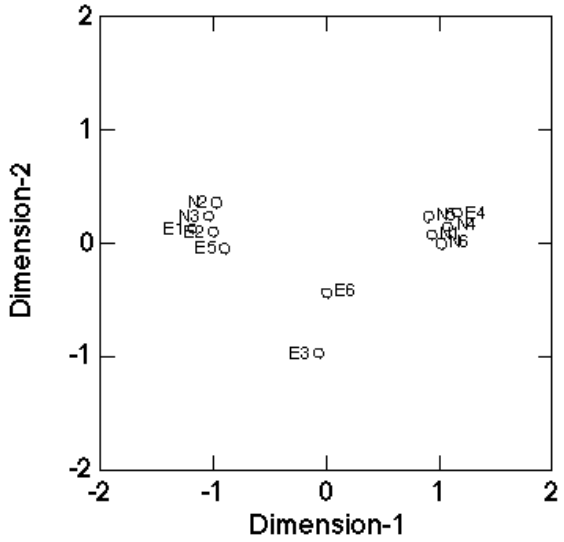

Fig. 5: MDS (r=1, Kruskal, Mono) plot with a binary normalised intersection matrix gives RSQ=0.995.

5.2.3 OUTCOME

According to the RSQ of Fig. 4 (RSQ=0.975) and of Fig. 5 (RSQ=0.995), we can explain most of the variance among user data. However, the binary based plot of Fig. 5 (RSQ=0.995) is slightly better than that of Fig. 4 (RSQ=0.975). That is surprising, since the method ignores information about repetitive behaviour. Maybe such information is redundant in the context of this method.

5.3 EXCLUSION METHOD

This method is based on the exclusion as a metric of difference. Exclusion among two users is always given by two areas. The area of one user STV (user 1, Fig. 2) excluded from the area of a second user STV (user 2, Fig. 2), is not the same as the area of the second user STV excluded from the area of the first one. Since the two exclusion areas are asymmetric, the method does not allow for MDS as grouping algorithm.

5.3.1 METRIC

This method measures the difference between two STVs by estimating how much of one user STV (column index in Table 2) is excluded from a second one (row index, Table 2).

Formula 3:

$$M_{p,q}^{EX} = \sum_{i=1}^{n} \left| \min\left(e_{p,i} - e_{q,i}, 0\right) \right|$$

where:

$M_{p,q}^{EX}$: Exclusion metric between user p and q

i : Summing Index STV elements

n : STV length, upper summing limit

$e_{p,i}$: STV element i for user p

$e_{q,i}$: STV element i for user q

Following this procedure for all users, we get an mxm asymmetrical matrix (Table 2), where each element is a measure of exclusion (Formula 3). Since there were six novices (N1-N6) and six experts (E1-E6), m is 6+6=12.

Table 2: Numerical representation of exclusion matrix.

E6	6	43	47	51	70	50	47	35	73	5	171	0
E5	17	15	14	69	48	67	23	7	64	21	0	24
E4	9	44	47	56	70	55	47	35	73	0	171	8
E3	17	41	41	68	77	62	28	28	0	21	162	24
E2	17	15	16	68	81	67	19	0	66	21	143	24
E1	20	16	19	73	85	72	0	7	54	21	147	24
N6	3	41	44	28	48	0	47	30	63	4	166	2
N5	3	39	42	41	0	33	45	29	63	4	132	7
N4	2	41	42	0	55	27	47	30	68	4	167	2
N3	16	11	0	68	82	69	19	4	67	21	138	24
N2	18	0	15	71	83	70	20	7	71	22	143	24
N1	0	41	43	55	70	55	47	32	70	10	168	10
	N1	**N2**	**N3**	**N4**	**N5**	**N6**	**E1**	**E2**	**E3**	**E4**	**E5**	**E6**

5.3.2 GROUPING ALGORITHM

The grayscale representation (Fig. 6) of the exclusion matrix (Table 2) is generated by Mathematica (Wolfram, 1991) *ListDensityPlot* with the negative, inverted exclusion matrix as input. We use the negative matrix to obtain a consistent plot. Fig. 6 is only meant as a visualisation of Table 2, and is not an exact mapping. Since division by zero is not defined, the diagonal elements of Table 2 were directly mapped to the darkest graytone. Fig. 6 shows to what degree a *column* user STV is excluded from a *row* user STV. Darker matrix elements correspond to lower degree of exclusion.

To interpret the degrees of exclusion in Table 2, we suggest an iterative *predictor-corrector* algorithm. The *corrector* is an estimator for the threshold value so that only considering exclusion measures between that value and zero will give the predicted number (*predictor*) of user groups. The stop criterion for the iteration method is that the number of user groups given by the corrector, equals

the value of the predictor. Research on converge criteria is part of our future work, so now we simply assume convergence. For each iteration the corrector is modified in order to meet the stop criterion, according to the following rules: If we consider too few exclusion relations (i.e. the corrector is too close to zero), the number of groups will be higher than the predictor. If we consider too many exclusion (i.e. the corrector is too far from zero), many or all of the users will be related by exclusion statements, and the number of groups will be lower than the predictor. We give our predictor the value predictor=3. By visual inspection of Fig. 6 it appears reasonable to consider the darkest matrix elements only. Since these elements have numerical values equal to or below 8 (Table 2), we choose the initial value of the corrector to be 8.

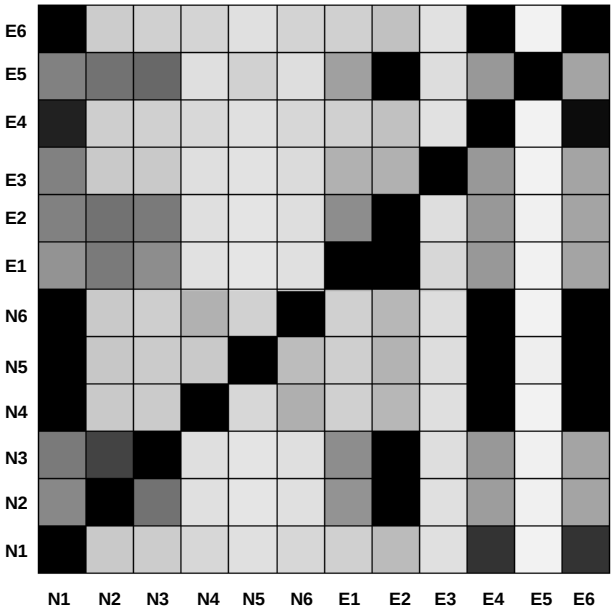

Fig. 6: Grayscale representation of exclusion matrix. Darker elements mean higher exclusion of *column* user STV from *row* user STV.

Diagonal elements are ignored, since each STV is fully similar to itself.

Since small differences indicate similarity, we can derive (based on Table 2) four similarity relations (Table 3).

Table 3: We can derive these four similarity relations.

Similarity relation	User STVs of each relation
1	N1 ∈ N4, N5, N6, E6
2	E4 ∈ N4, N5, N6, E6
3	E6 ∈ N4, N5, N6, E4
4	E2 ∈ N2, N3, E1, E5

All users that are related by an similarity relation are defined to belong to one group. Since the three first similarity relations (Table 3) are interrelated, this gives one group. The remaining, fourth similarity relation (Table 3) gives a second group. Users not appearing in any similarity relation define a separate group.

5.3.3 OUTCOME

Hence, the algorithm gives the following groups: {N1, N4, N5, N6, E4, E6}, {N2, N3, E1, E2, E5} and {E3}. We assumed that the number of groups should be three, so the stop criterion has already been met. If our prediction had not been met, we would have to try with a higher or lower corrector (according to the above mentioned rules) and go back to the start of the predictor- corrector algorithm. This algorithm is repeated until the stop criterion is met (convergence).

6 DISCUSSION

In order to validate the outcome of these three automatic methods, we performed a protocol analysis (Ericsson and Simon, 1984) of the task. This is manual work, based on analysis of video and verbal utterances in addition to logfiles. This is mostly feasible for simple tasks, where users basically follow one or a few strategies. This analysis showed that there are three distinct strategies solving the task. We call these strategies S1, S2 and S3. Table 4 shows the users according to their successful strategy.

Table 4: Manual protocol analysis of the task shows three distinct strategies and gives information about which user succeeded by which strategy.

Strategy	Users according to strategy
S1	N1, N4, N5, N6, E4, E6
S2	N2, N3, E1, E2, E5
S3	E3

The strategies are represented as STVs and have the same qualitative interpretation as the STVs of the users (N1-N6) and (E1-E6). We see that the correlation method and intersection method do not correspond fully with the outcome of the protocol analysis. The exclusion method, however, gives exactly the same results. So, the exclusion method is the best one with our combination of system, task and users behaviour. In the future, we want to find out how the different methods, especially the exclusion method, perform with other, more complex tasks.

We have seen that for a relatively simple task, the method which is purely analytical (exclusion method) is the best one. Measured by the RSQ-values, the intersection method is better than the correlation method, which is purely statistical. This indicates that in our context, statistical methods offer less explaining power than the analytical methods for strategy and plan recognition.

7 CONCLUSION AND FUTURE PERSPECTIVES

We have acquired results for one task only. To make our methods more reliable, we need to evaluate several tasks. For each task, we will validate our methods by manual protocol analysis.

We also plan to study learning experiments, in order to recognise the acquisition process of strategies.

M. Fjeld, S. Schluep & M. Rauterberg (1998): Automatic, action driven classification of problem solving strategies by statistical and analytical methods: a comparative study. In F. E. Ritter & R. M. Young (eds.) Proceedings of the Second European Conference on Cognitive Modelling, pp. 98-103. Nottingham: Nottingham University Press.

8 REFERENCES

Ericsson, K. A., & Simon H. A. (1984). Protocol analysis, verbal reports as data. The MIT Press.

Rauterberg, M. (1992). An empirical comparison of menu selection (CUI) and desktop (GUI) computer programs carried out by beginners and experts. *Behaviour and Information Technology 11*, pp. 227-236.

Rauterberg, M. (1996). A Petri net based analyzing and modelling tool kit for logfiles in human computer interaction. In (Yoshikawa, H., & Hollnagel. E., eds.) *Proceedings 'Cognitive Systems Engineering in Process ControlSCEPC'96*. Kyoto University, pp. 268-275.

Rauterberg, M., & Fjeld, M., & Schluep, S. (1997). Parallel or event driven goal setting mechanism in Petri net based models of expert decision behaviour. In (Bagnara, S., & Hollnagel, E., & Mariani, M., & Norros, L., eds.) *Proceedings of CSPAC'97*. CNR, Roma, pp. 98-102.

Ritter, F. E., & Larkin, J. H., (1994). Developing Process Models as Summaries of HCI Action Sequences. *Human Computer Interaction* 9, pp. 345-383.

SYSTAT Inc. (1989). SYSTAT®: The system for statistics. pp 93-166. SYSTAT program version 7.0.1 for PC.

Tauber, M. J., (1985). Top down design of human-computer systems from the demands of human cognition to the virtual machine - an interdisciplinary approach to model interfaces in human-computer interaction. In Proceedings of the IEEE workshop on languages for automation (Palma De Mallorca (E) June 28-29), pp. 132-140.

Wolfram, S. (1991). Mathematica®, A system for Doing Mathematics by Computer. 2nd Ed., AddisonWesley, pp. 164, 395, 819. Mathematica program version X3.0.1.1 for Silicon Graphics IRIX.