Cogn Tech Work (2005) 7: 241–261
DOI 10.1007/s10111-005-0010-z

ORIGINAL ARTICLE

Tom Kontogiannis

Integration of task networks and cognitive user models using coloured Petri nets and its application to job design for safety and productivity

Received: 6 May 2004 / Revised: 7 February 2005 / Accepted: 5 June 2005 / Published online: 9 November 2005

Abstract Changes of task demands due to unforeseen events and technological changes can cause variations in job design such as modifications to job procedures and task allocation. Failure to adapt to job design variations can lead to human errors that may have severe consequences for system safety. Existing techniques for task modelling cannot adequately model how task networks can be adapted to changing work conditions and task demands. Therefore, there is a need to integrate task networks with cognitive user models that indicate how operators process information, make decisions, or cope with suspended tasks and errors. The work described here presents a tool for integrating task and cognitive models using coloured Petri nets. The cognitive user model comprises two modules of attention management (selective and divided attention), a module of memory management of suspended tasks and a module of work organization. Performance Shaping Factors (e.g., workload, fatigue and mental-tracking load) are calculated at any point in time to take into account the context of work (e.g., competing activities, errors and suspended tasks). Different types of human error can be modelled for rule-based behaviours required in proceduralized work environments. Simulation analysis and formal analysis techniques can be applied to process control tasks to verify job procedures, workload management strategies and task allocation schemes in response to technological changes and unfamiliar events.

Keywords Task networks · Cognitive modelling · Human reliability · Job design · Safety · Coloured Petri nets

T. Kontogiannis
Department of Production Engineering and Management, Technical University of Crete, University Campus, Chania, Crete 73100, Greece
E-mail: konto@dpem.tuc.gr

1 Introduction

Many reviews of near miss and incident analysis in process control have identified a common pattern of operational problems that are related to how people adapt to variations in job design caused by unforeseen events and technological changes (Kletz 1991; Hollnagel 1993; Embrey et al. 1994; Kjellen 2000). Operational problems may include machine failures, unavailability of tools, operation of new or unfamiliar equipment, encountering high levels of workload, carrying out the same job with fewer people, and so forth. In such cases, operators may have to modify their goal priorities, job procedures, allocation of tasks and use of tools or resources. These modifications may initially lead to unsuccessful actions as they are often carried out under time pressure and frequent interruptions. The ability of job design to adapt to variations in ordinary practice and recover from early unsuccessful actions will affect not only productivity but also system safety.

Matching job requirements to human capabilities has been the concern of task analysis for several decades and a range of techniques have been developed in the domain of industrial ergonomics (Kirwan and Ainsworth 1992; Stanton and Young 1999). Task analysis techniques have been successfully used to describe operator goals, allocation of tasks and scheduling of tasks to optimize man-machine interaction. However, task analysis describes 'how the job should be done' under a set of well-defined conditions of the work environment. In this sense, it runs into difficulties in cases where technological changes and unanticipated events require variations from ordinary practice. The effects of job variations, such as modifications in procedures and changes in task allocation, are usually examined at a later stage with the use of a complementary human error analysis. Another concern with traditional task analysis is the design of new jobs for which little information exists for the staffing levels and the required scheduling of tasks and resources. It appears that new techniques are needed

that would examine systematically variations in existing jobs as well as model requirements of new job designs.

The aim of the work described here is to develop a tool for generating task networks and modelling rule-based behaviour in work environments characterized by frequent variations in procedures and task allocation. The output of task analysis and additional data are used to specify a task network which models the flow of information, the utilization of resources and the control of tasks. This task model is integrated with a cognitive user model that examines strategies for changing goal priorities, coping with suspended tasks, managing workload, and allocating resources to people. Hence, the cognitive user model can be used to adapt an ordinary task network to the changing conditions of the work environment. An error taxonomy (Hollnagel 1993) has been used to model the consequences of human error and the potential for error recovery. A computer simulation tool has been developed in order to cope with the large amount of data required to carry out an extensive analysis of tasks in proceduralized work environments and to run several combinations of work conditions that may lead to human errors due to inefficient strategies in managing workload and task allocation. In this way, the design of existing and new jobs can be verified in terms of cognitive user strategies and standard operating procedures.

Previous approaches to task modelling have tended to focus mainly on computer representations of operator tasks and plans, without any elaboration on the cognitive strategies employed by human operators. Task modelling tools, such as MicroSaint (Laughery and Corker 1997) and GOMS (John and Kieras 1996), provide a useful basis for examining the effects of several policies for task allocation and scheduling on the design of procedures but do not incorporate any user models to indicate how operators process information, make decisions, or cope with suspended tasks and errors. In the context of human-computer interaction, task modelling tools - such as ConcurTaskTrees (Mori et al. 2002), TOBOLA (Uhr 2003), TOOD (Abed et al. 2003) and DIANE+ (Tarby and Barthet 1996) - have been developed to design the user interface but they provide limited facilities for modelling cognitive user strategies.

The benefits of integrating task models and cognitive user models are clear. First, the way that operators adapt to variations in job procedures imposed by technological changes can be studied better with cognitive user models that address attention management and decision-making with regard to changing work conditions, goal priorities, and scheduling requirements. Secondly, coping with new events, task interruptions and unsuccessful attempts requires human memory models of how people keep a mental track of their activities and their own progress. Thirdly, cognitive user models provide a good basis for estimating the dynamic effects of performance shaping factors (PSF) on human reliability (e.g., models of divided attention have been used to model workload). Finally, the consequences of human error can be better understood when cognitive user models specify how these errors arose in the first place.

In view of these benefits, an interest has been witnessed increasingly in recent years in tools that integrate task models with cognitive user models. In the context of human computer interaction, for instance, Lebiere et al. (2002) integrated a task modelling tool (i.e., IMPRINT) with a widely acceptable cognitive model of human memory and learning (i.e., ACT-R). In the area of aircraft maintenance, Cacciabue et al. (2003) developed a simulator tool that integrated task modelling with a user model that estimated PSF dynamically. Other developments in human reliability focused on formal models of user behaviour that predicted human errors in the context of air traffic control (Lindsay and Connelly 2002). Finally, a generic purpose tool was developed by Hendy and Farrell (1997) that integrated Perceptual Control Theory with task networks for a wide variety of applications.

Petri nets have been used in several applications as a common technique for the modelling of tasks, users and technical systems. An example is the work of Palanque and Bastide (1997) and Lacaze et al. (2002) which integrated a formal representation of the system with the Human Processor Model of Card et al. (1983). Other studies have used Petri nets modelling to reason about users' mental models and their expectations over the course of an interaction (Rauterberg 1993; Moher and Dirda 1995). In recent years, Petri nets provided the basis for the analysis of human interactions with complex industrial systems. In the nuclear power sector, for instance, Yoshikawa et al. (1997) used Petri nets to develop a cognitive simulator model of how operators diagnose accident sequences. A study by Stutz and Onken (2001) provided useful insights into the adaptation of normative pilot models based on Petri nets with the use of case-based reasoning while Blom et al. (2000) looked into aspects of situation assessment in air traffic control systems using Dynamic Stochastic Petri Nets. Finally, in the defense sector Handley and Levis (2003) used coloured Petri nets (CPN) to integrate decision-making and operational planning in command and control tasks.

The modelling approach that is proposed in this work is based on a Petri net representation of human activities, tools, and organizational roles. Petri nets are cast both in a graphical form and a mathematical formalism. The graphical form of Petri nets helps to visualize state transitions in cognitive processes where both serial and parallel processing can be mixed. Although most of the modelling work can be done in the Petri nets graph, the mathematical foundation provides the basis for using a variety of formal analysis techniques. Formal analysis techniques can be built into software packages in order to examine several structural and dynamic properties of Petri nets. Formal analysis uses reachability graphs and linear algebra to examine whether a task network contains deadlocks, livelocks (i.e., never-ending tasks), dangling tasks that do not contribute to the goal, dead

tasks that never occur, and tasks that result in lack of synchronization. The reachability graph can be used to examine certain behavioural properties (i.e., liveness) and identify routes of tasks that may lead to 'hazardous' states (Levenson and Stolzy 1987). Linear algebra can enable analysts to prove certain structural properties in terms of place and transition invariants. For instance, place invariants may prove that a task can be found in a certain number of states or that certain tools are used throughout a job. On the other hand, transition invariants can be used to examine how a system can be decomposed into subsystems that perform dedicated functions, such as an autonomous cyclic processing of data and products. Finally, Petri nets provide a common language for modelling both the technical system and the human element. In this way, research done in the formal analysis of manufacturing systems may be explored in the modelling of man-machine systems. The proposed modelling framework is based on coloured Petri nets CPN implemented on the technical platform DesignCPN (Jensen 1997a, b).

The paper is structured as follows. First, a framework is proposed for integrating task networks with cognitive user models which presents the flow of decisions and data as well as the interactions between the constituent parts. The following section focuses on the theoretical structure adopted for representing human performance in the context of existing cognitive modelling approaches. The computer implementation of the task networks and cognitive models in terms of CPN is then described in detail. A case study is presented to illustrate the capabilities of the simulation analysis tool in the context of a process control task. State space analysis of the CPN model is not examined as this was beyond the scope of the current work. Finally, the capabilities and limitations of the simulation tool are discussed in the concluding section.

2 A framework for integrating task and user models with coloured Petri nets (CPN)

The proposed framework consists of a task model or network that specifies the organization of tasks required to achieve certain goals and a cognitive user model that specifies how operators select and prioritize tasks, how they allocate tasks according to their preferences and job demands, and how they keep mental track of suspended or failed tasks. An executive mechanism is used that monitors external events and makes decisions about changes in task processing. The task network, the user model and the executive mechanism are being executed in parallel (Fig. 1).

A key construct in this framework is *tasks*—i.e., the building blocks of operations required to achieve certain goals. Tasks can be scheduled and organized in different ways to form higher-order operations as well as they can be served by different operators and material resources. Tasks are characterized by certain attributes associated with their temporal order such as, pre-conditions, task

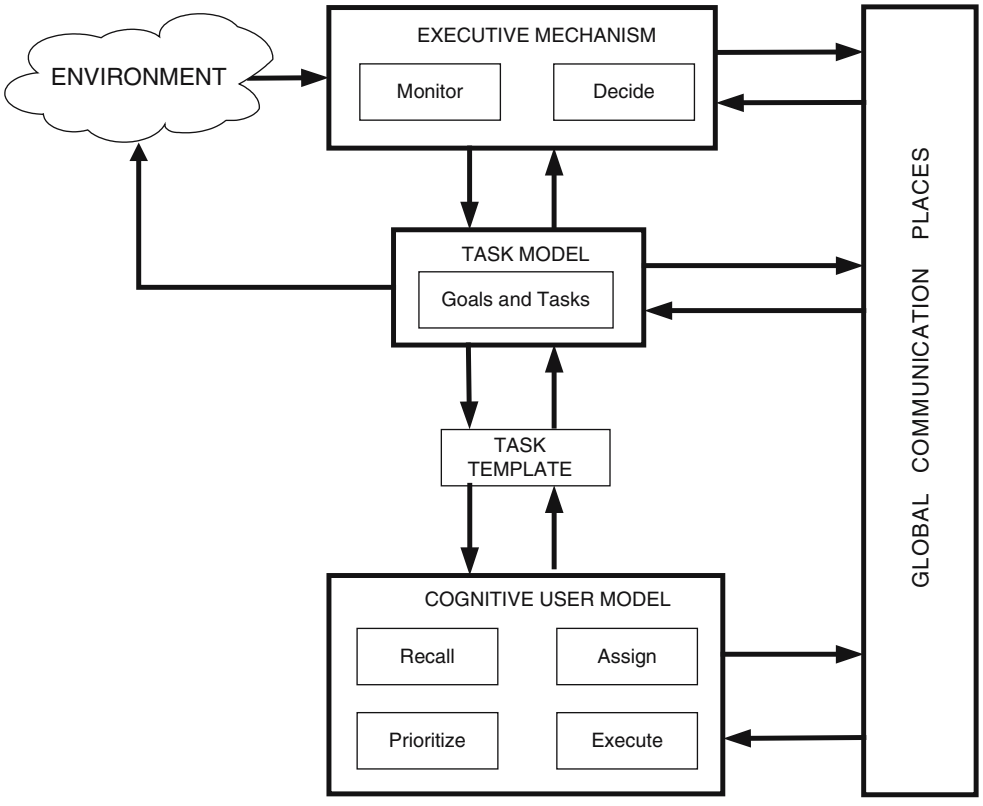

Fig. 1 A proposed framework for integrating task models and cognitive user models. Monitoring and decision-making are modelled separately in the executive mechanism

durations, task priorities and primary or contingent operators. There are also additional task attributes related to particular aspects of the cognitive user model. For instance, tasks are rated in terms of the degree to which they rely on perceptual, cognitive and motor channels in order to estimate the workload of operators when attention is divided between multiple tasks. Other task attributes may be related to the external cues that remind operators to resume suspended tasks in the memory module of the user model. In this sense, a large amount of data is required as different task attributes are associated with different aspects of the task network and the cognitive user model.

The task model is a network, or organization of tasks, specifying the normative sequence of tasks as described in the operating procedures. Alternative sequences can be tested in the task model in order to find out the most efficient ones, or to examine the consequences of human error in planning activities. Information from the task model is passed on to the cognitive user model through a task template which transforms the graphical form of a task into a *token*, or mark, that enters the user model for further processing (Fig. 1). Petri nets enforce a 'point-to-point' or local communication framework which, on some occasions, may restrict the management of dynamic changes to data and tasks. For this reason, a small number of global fusion places were created to enable some degree of global communication between Petri nets modules.

The user model comprises four cognitive models that address issues of task selection, mental tracking (e.g., recalling suspended tasks), operator assignment and execution of tasks. Human errors are modelled either at the stage of planning sequences in the task model, or at the stage of execution in the user model. The consequences of errors are specified in the execution module and are spread throughout the task network. The interactions between task network, cognitive user model and executive mechanism are best seen in the data and decision flowchart (Fig. 2). The paths taken through this flowchart describe the data and decisions made by several CPN models in a way that is easier to communicate. The decision blocks of the flowchart correspond to smaller CPN models that collect data, make decisions and produce outputs. Ovals depict queues of tasks that are locally accessible (i.e., local communication) while cylinders depict databases holding dynamic information about the state of tasks that are globally accessible (i.e., global communication).

The executive mechanism initiates the task network which sends new tasks to the cognitive user model according to a specified plan or sequence of tasks. The tasks are seen as discrete units that may be skipped or forwarded for processing in the user model. A list of candidate tasks is first formed that includes tasks being suspended in the *Suspended_Tasks_Memory* place. A recall/shed algorithm is used to determine tasks that may be forgotten due to an excessive load in mental tracking. Candidate tasks are prioritized and join a list of selected tasks for further processing or await selection for longer periods of time. Prioritization of tasks is based on another algorithm derived from a model of selective attention. Selected tasks are assigned to operators according to the allocation rules and operator preferences specified in a work organization model. Parallel processing of tasks is affected by the workload of operators as calculated by a fourth algorithm based on a model of divided attention. Ongoing tasks that may be executed in parallel can join the list of active tasks for execution. Six workload-management strategies are implemented to cope with task allocation and the processed tasks are subsequently executed in the user model.

The output of the cognitive user model (i.e., correct actions and human errors) is sent to two databases holding information about the state of current tasks and suspended tasks. Several performance problems can be modelled including omissions, mistiming, interruptions and delays. Commission errors are modelled separately as planning errors in the task model. Tracking errors can also be considered when suspended tasks are shed as a result of high load in mental tracking. Errors can have consequences on the performance of tasks and may cause side-effects to other tasks performed earlier in the sequence. The decision model monitors the work environment and determines whether the current goal should be interrupted or not, in favour of others. In addition, the decision model can initiate the recovery of tasks that have been upset, shed, or failed earlier.

3 Cognitive modelling

In recent years, there has been a growing effort in developing computer models of human performance as summarized in several revisions of the literature (e.g., Cacciabue 1998; Pew and Mavor 1998; Zachary et al. 1998). Some studies have focused on detailed representations of human cognition while others emphasized the interaction between cognition and work environment. This time-dependent interaction between cognition and work is very important for the study of human reliability in complex work environments. The selection of models of attention and memory in this work was guided by the extent to which candidate models made reference to the context of work (e.g., current tasks, competitor tasks, suspended tasks and failed tasks) and included task characteristics that could be changed as part of the job-redesign process. Some of the selected models of attention and memory had been simplified to enable quantification of the man-machine interaction in a way that was appropriate to the modelling language. Petri nets is a tool that has been used primarily for modelling technical systems, however, there is some potential to be explored for modelling human performance as well (Blom et al. 2000; Schlick et al. 2002; Handley and Levis 2003).

The cognitive user model includes a work organization module for assigning tasks to operators according to their capabilities, preferences and system allocation

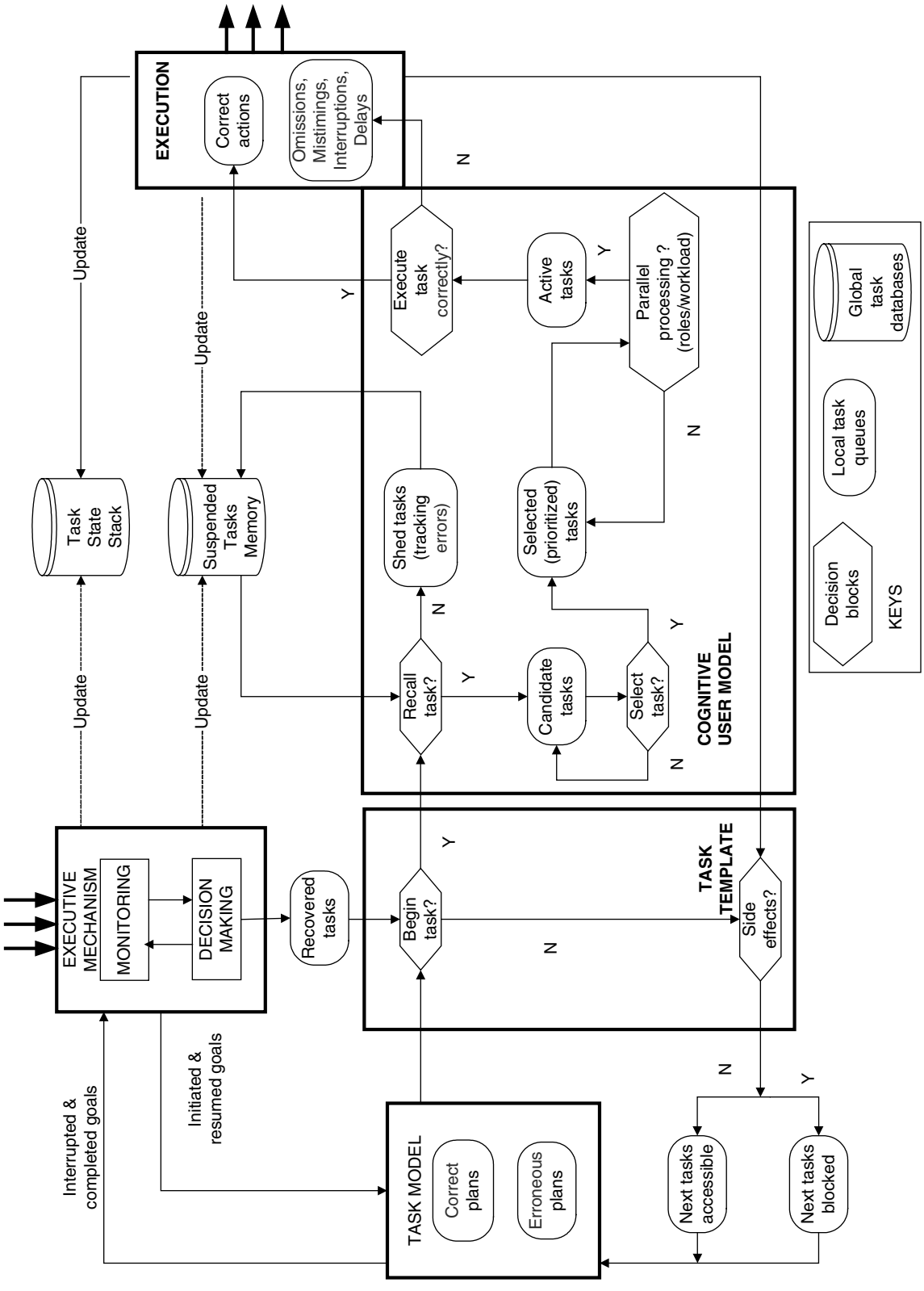

Fig. 2 A flowchart showing the flow of data and decisions in the proposed framework

rules. Monitoring and decision-making are two cognitive processes that have not been adequately developed in the present version of the simulation tool. As they were not based on formal models of cognition, they were not included in the cognitive user model. However, they were considered in the executive mechanism that run parallel to the modules of attention and memory management.

3.1 Attention management

The purpose of the attention management module is to schedule tasks to be performed, taking into account certain priorities, task constrains and interferences at any point in time. This module determines whether a task is performed on demand, interrupted, resumed or postponed. The task network generates new tasks that may compete for attention with tasks actively running, or tasks being temporarily suspended. When a new task arrives in the cognitive user model, a list of candidate tasks is made from new tasks and suspended tasks that could be possibly resumed. Selective attention is a dynamic process that considers several priorities for selecting which tasks to consider first. Although tasks have their own default priorities, the selective attention module computes a *priority index* that takes into account the context of work at any point in time (e.g., other competing tasks).

The following computing algorithm has been adapted from previous work by Freed (1998, 2000):

$$P = \mathrm{IC} + \sum_{i=1}^{n} (p_i \cdot c_i) \tag{1}$$

where

P is the priority index for a new or suspended task,

- IC is the interruption cost associated with interrupting or deferring a task,
- $i = 1 \ldots, n$ potential consequences as a result of deferring a task,
- p_i is the probability of incurring the ith consequence, and
- c_i is the cost of the ith consequence

The interruption cost (IC in Eq. 1) is a numerical value that incorporates the cost of interrupting or deferring a task. It is associated with maintaining tools and resources in appropriate states during the interruption interval and re-establishing preconditions in order to reconsider the task at a later stage. The second term of Eq. (1) refers to the risk of incurring an adverse consequence as a result of interrupting or deferring a task. The idea of a deadline, after which undesirable consequences may occur, is a central feature of any approach to prioritization (Freed 2000). Deferring a task from selection may be associated with several deadlines, or consequences, that have different probabilities of occurrence. This probability is a function of several factors including, (1) the time remaining until a consequence is ensued, (2) the duration of the task, (3) the times the task has been interrupted or suspended in the past, and (4) the characteristics of current competitor tasks. The last factor is the most difficult to calculate because it requires predicting what competitor tasks may exist at a second or third attempt to select a task that has been deferred in the past. This ability to 'project into the future' and predict likely competitor tasks is not elaborated in the present module which considers competitor tasks in the current attempt only. The product of probability and cost is the risk of a consequence that is summed up with other consequences. This sum is added to the interruption cost in order to calculate *a priority index* for a new or suspended task. The 'selective attention' module selects the task with the higher priority index which is added to the list of tasks awaiting further processing (see list of selected tasks in Fig.2). Tasks with the same priority are scheduled according to their originally scheduled start time or least task duration.

A second attention mechanism operates on the selected or prioritized tasks in order to keep operator workload within acceptable tolerance limits. Tasks that share the same information processing channels will increase the workload and, hence, compete for processing time. A 'divided attention' module operates in order to calculate interference levels arising from parallel processing of tasks. A *workload index* is used to estimate the interference level on the basis of the Multiple Resource Model (MRM) of Wickens et al. (1988). Each task is rated in terms of the demands on the perceptual, cognitive and motor channels and the combined effects are used in computing a workload index:

$$W_T = \sum_{i=1}^{m} \sum_{t=1}^{n} (a_{t,i}) + \sum_{i=1}^{m} c_i \sum_{t=1}^{n-1} \sum_{s=t+1}^{n} (a_{t,i} + a_{s,i}) \tag{2}$$

Where

- W_T is the instantaneous operator workload at time T,
- $i = 1, \ldots, m$ resource channels,
- $t,s = 1, \ldots, n$ competing tasks,
- $a_{t,i}$ is the loading on channel i to perform task t,
- $a_{s,i}$ is the loading on channel i to perform task s, and
- c_i is the conflict between tasks in sharing channel i

The first term of Eq. (2) is the first order workload component and represents an aggregate of concurrent task demands at a given point in time. The second order component represents conflict due to tasks sharing the same information channels. It calculates the demand values within each channel and multiplies the resulting sums by a conflict value corresponding to the interference level in the channel in question. This is done for each channel separately and the results of intra-conflict are summed up for all information-processing channels. Inter-conflict between channels is not considered in this module but it is possible to calculate on the basis of the algorithm suggested by North and Riley (1989).

The scheduling of tasks is influenced by the strategies that operators use to manage their workload at different points in time. This module includes the following strategies that can be triggered when an operator's workload exceeds a pre-defined threshold value (Lockett 1997):

A. A penalty is incurred in terms of reduced accuracy or speed of execution
B. The new task is not started by any other operator which is referred to as 'deferring or skipping the task'
C. The new task is performed in serial rather than in parallel to ongoing tasks
D. The new task is reallocated to another operator who is less busy
E. An ongoing task is reallocated to a new operator
F. An ongoing task is interrupted in favour of processing the new task

Strategies for managing operator workload can be defined by forming combinations of the six basic strategies. Therefore, the scheduling of tasks will be determined by the momentary workload of operators and the selected strategy for managing workload. Workload is supposed to follow an exponentially decay function and its accumulated effects are taken into account in an *index of fatigue*. In this sense, workload and fatigue can be viewed as two PSF that impact upon the likelihood of human error.

3.2 Memory management

A temporary task queue is generated and regulated according to a model of human memory in order to manage the large number of new candidate tasks, suspended tasks, tasks awaiting processing (i.e., the list of selected tasks) and ongoing tasks (i.e., the list of active tasks). The activation-based model of memory (Altman and Trafton 2002; Lovett et al. 2000) has been adapted which views retrieval of suspended tasks from memory as a function of the 'base-level activation' specific to the task (internal cueing) and the 'external activation' received from task cues in the work environment (external cueing). The 'base-level activation' depends on the importance of the task and its frequency of rehearsal in memory. On the other hand, external cueing depends on the number and strength of task cues that may remind operators of a suspended task pending execution. Hence, each suspended task acquires some degree of activation in the working memory and the task can be recalled when its activation level exceeds a threshold. A tentative algorithm for modelling the activation level of suspended tasks has been developed by building on earlier models of Lovett et al. (2000). The proposed algorithm for calculating the activation level (A) of suspended tasks is as follows:

$$A = \mathrm{FC} + \left(\frac{\mathrm{TA}}{n}\right) \cdot \frac{(1 - W_c)}{W_{\max}} \cdot \sum_{i=1}^{m} \frac{S_i}{m} \qquad (3)$$

where,

- A is the activation level of a suspended task,
- FC is the cost of forgetting a suspended task,
- TA is the total activation level of attention spreading to n tasks,
- n is the number of tasks in the focus of attention (i.e., new tasks, suspended tasks, prioritized tasks and ongoing tasks),
- w_c is the current workload of operator,
- $w_{\max}$ is the maximum workload value of operator,
- s_i is the strength of recall of the ith external cue, and,
- $i = 1 \ldots m$ external cues for a suspended task.

The first term of Eq. (3) refers to the 'base-level activation' of a suspended task and has a numerical value corresponding to its importance or cost of forgetting. The second term describes the dynamics of external cueing in human memory. The total activation level (TA) is spread equally over all tasks and external cues that are currently in the focus of attention. In general, a task will be more active, the more strongly related it is to a set of external cues (i.e., the sum of s_i of cues) which corresponds to data-driven processing. Top–down processing can be modelled by the portion of attention allocated to each task when other tasks compete for attention. To model top-down processing, the TA is divided by the number of tasks (n) in the focus of attention (i.e., new tasks, suspended tasks, selected tasks and ongoing tasks); it is also assumed that only a potion of total activation will be available to all tasks due to the current workload of operators. From a cognitive science perspective, it is very interesting to examine the relative values of the two terms of Eq. (3) since a high value of 'base-level activation' will render the activation level (A) insensitive to the context of work. In other words, an important task has a high 'base-level activation' (i.e., it is frequently rehearsed) and does not need many external cues to remind operators when to reconsider this task in future.

The number of tasks in the focus of attention (n) can be seen as an index of the memory load in tracking the state of various tasks and remembering their implications for action. It is likely that the type of task and its implication for action will affect memory load in tracking progress. In the current version of the tool, the number of tasks in the temporary queue (n) is used as a simple *index of mental tracking load*. It is worth noting that the *workload index* refers to ongoing tasks that are currently running and does not include suspended tasks and selected tasks awaiting processing. Therefore, the index of mental tracking load can be seen as a separate performance shaping factors (PSF).

3.3 Work organization

To model the performance of a crew of operators it is necessary to have a module of work organization that specifies how the work is allocated to different operators

according to certain allocation rules, operator capabilities and workload levels. Allocation rules may be specific to a few tasks (e.g., two tasks should be done by the same operator), or may concern generic strategies for task allocation. For instance, a global rule may specify that an operator is not allowed to do more than four tasks when someone else did nothing during the same period of time. Therefore, it is possible to have a *compliance score* for each operator describing the extent to which his or her performance complies to a local or global rule. The capabilities of operators can be taken into account by incorporating their accuracy and speed of performing a task in an algorithm that computes an assignment score (R) for each operator at a point where a decision must be made on how to allocate a task.

The following algorithm has been proposed to compute the assignment score for each candidate operator at any point in time:

$$R = P \times (A/D) \cdot \prod_{i=1}^{n} C_i \quad (4)$$

where,

- R is the assignment score for an operator,
- P is the priority value of a specific task,
- A is the task accuracy val ue achieved by an operator
- D is the task duration value achieved by an operator
- $i = 1, ..., n$ rules for task allocation, and,
- C_i is the operator compliance score with the ith allocation rule.

As argued before, tasks can have certain priority values that change over time depending on the context of work. Equation (4) uses a simpler expression for the priority value of tasks by considering only the default priority of tasks (i.e., the standard cost of interruption as shown in Eq. 1). The compliance score of each operator to each allocation rule can be taken by the expression $C_i = 1 - \text{Penalty}(i)$, where a penalty value is assigned to any violations of the rule. The algorithm in Eq. (4) selects the operator with the highest assignment score as the most suitable candidate to perform a specific task. If the selected operator had a high workload level, then the next operator is selected with the lower assignment score.

4 CPN modelling

4.1 Basic concepts

Coloured Petri nets (Jensen 1997a, b) have been chosen as a candidate for modelling because they provide facilities for hierarchical and timed descriptions, communication of 'data structures' (i.e., Coloured tokens), simulation of tasks, and formal analysis of reachability or occurrence graphs. Task analysis can be carried out by decomposing tasks into hierarchies of operations and plans in a manner similar to hierarchical task analysis (Shepherd 2001). For this reason, a classification of plans has been developed for Petri net-based descriptions of human plans and task sequences (Kontogiannis 2003). Structural and dynamic properties of the task network are investigated with the use of occurrence graphs that are automatically constructed by the DesignCPN software package. The simulation facility of DesignCPN is used for collecting performance data, such as time to perform tasks, idle time for operators and machines, product throughput, and operator workload. In addition, task simulation allows analysts to explore a variety of questions related to aspects of plan implementation. Some of those questions include the following: are there enough resources (human operators and tools) to achieve a plan? How long will it take to achieve a plan? Are there any periods where some operators are overloaded or waiting idle? What deadlocks should be avoided to ensure that the goal is achieved? What activities should be done in parallel to speed-up a task without creating deadlocks? These questions guided the current work in modelling task activities with Petri nets.

Formally, the structure of a Petri net is a bipartite directed graph $G = (P,T,A)$, where $P = \{p1, p2, p3, ..., pn\}$ is a finite set of places, $T = \{t1, t2, t3, ..., tn\}$ is a finite set of transitions, and $A \subseteq (P \times T) \cup (T \times P)$ is a set of directed arcs. The set of input places of a transition (t) is given by $I(t) = \{p|(p,t) \in A\}$ and the set of output places is given by $O(t) = \{p|(t,p) \in A\}$. A high-level Petri net graph comprises the following elements:

- *A Net graph*. It consists of nodes (i.e., places and transitions) and arcs connecting places to transitions.
- *Places*. They are passive elements that can describe initial and end states of tasks, preconditions, and buffers for holding tools and resources. Places are represented as circles or ovals.
- *Tokens*. Tokes are 'data items' that reside in places and move throughout the network. Each place can hold several tokens of the same data type which are usually depicted as small and solid circles.
- *Arc annotations*. Arcs are inscribed with expressions that may comprise constants, variables and functions. The expressions are evaluated by substituting values for the variables. Inscriptions on transitions (i.e., *guards*) evaluate to Boolean values.
- *Transitions*. They are active elements that can be used for representing tasks, transportation of resources, routing of tasks and processing of events. Many software packages allow designers to write codes inside the transitions to enable control of the flow of tokens and evaluation of arc annotations.
- *Declarations*. They are statements regarding the data types held by places, the types of variables, and the specification of several functions.

A CPN model of the system describes the states in which the system may be found and the transitions between these states. The state of the system corresponds

to the distribution of tokens to all places at any point in time. When a transition is activated or fired, tokens are consumed in the input places and new tokens are created in the output places, according to the rules specified in the arcs or the transition codes. A CPN model has an initial state and may have several end states which represent desired and undesired outputs of the model. There can be several hundred or thousand paths between the initial state and the end states and this set of paths constitutes the state space of the CPN model. The software tool DesignCPN contains formal analysis methods which can be used to analyse the state space of the model and find efficient and problematic paths between inputs and outputs. A simulation corresponds to a single path of the state space and cannot be trusted to search all possible paths. Formal analysis techniques allow the analysis of a CPN model without the need to conduct simulation runs. However, if one is interested in the statistical significance of certain variables in a model, simulation runs will then be necessary.

4.2 The task model

The task model describes the normative sequence of tasks which can be adapted to the changing work conditions in two modes: (1) *online adaptation* via the cognitive user model and (2) *offline adaptation* by changing the structure of the task model. For instance, adjusting the start or duration of tasks, changing the order of tasks, and switching between serial and parallel processing can be made in the on-line mode. On the other hand, carrying out new sequences of tasks can only be done off-line because this requires some changes in the organization of the task network. The reason for the off-line adaptation is that the task model was cast as a tree graph (Fig. 3) in order to keep it compatible with existing methods of task analysis. Kontogiannis (2003) developed a Petri net notation of fifteen plans or sequences of tasks that could match the plans encountered in hierarchical task analysis (examples can be seen in

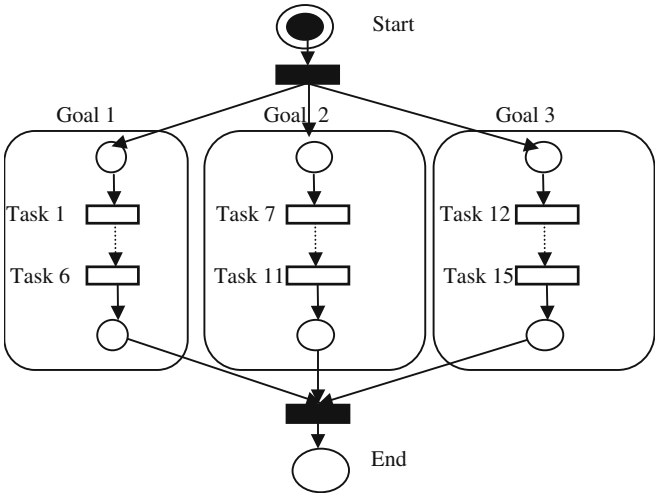

Fig. 3 An example of a task model showing a hierarchical job description in terms of goals and tasks

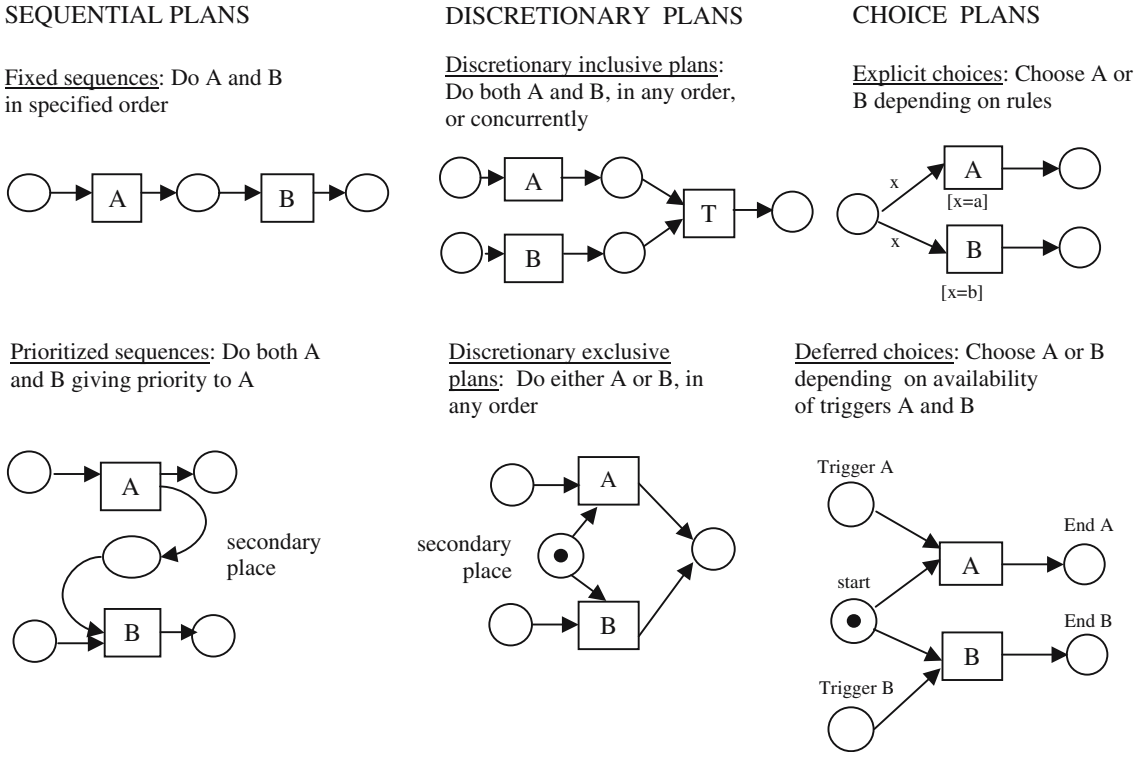

Fig. 4 A Petri net notation of different types of task sequences (Kontogiannis 2003)

Fig. 4). In fact, task modelling allows a more precise definition of plans than task analysis does, in a notation that is also executable. For a thorough presentation of workflow patterns and their transformation to Petri nets, the reader is referred to van der Aalst and Hofstede (2002).

Tasks are represented as transitions (white boxes in Fig. 3) with input and output places whilst other transitions may be used to create different task routings that correspond to particular plans of action (black boxes in Fig. 3). Places hold information about task attributes and task triggers (circles or ovals in Fig. 3) while tokens circulate this information throughout the network (small solid black circles in Fig 3). Task sequences are organized so that they form different 'wholes' corresponding to operator goals; hence, a hierarchical organization of goals is produced (Fig. 3). Some CPN models have used a 'tokens as tasks' perspective because this enables dynamic insertion and deletion of tasks (e.g., Handley et al. 1999; Petrucci et al. 2002); however, this architecture may lose the visualization of task sequences that is shown in the 'transitions as tasks' perspective (Figs. 3, 4). In order to increase the degree of on-line adaptation, the proposed framework transforms the tasks into tokens at a later stage, as they enter the cognitive user model.

This is achieved with the use of a task template (as shown in Fig. 5) which decomposes further each 'task transition' into a set of places and transitions. The routing of the token from the task model towards the user model is determined by the code segments in transitions *Begin_Task* and *End_Task*. At this stage, the 'task token' contains information about the identity of the task and the goal only, whereas additional attributes are considered in the cognitive user model. A task can be found at eight states that have implications for further processing (e.g., default, started, done, interrupted, deferred, failed, upset and shed). Information about states and other task attributes is held in a global-fusion place *Task_State_List* that is updated by the user model.

Goals can be interrupted and resumed by the executive mechanism which monitors the state of the technical system and makes decisions about goal planning. Interrupting a goal is indicated by putting its flag down and this results in suspending all uncompleted tasks (see arc with inscription *flag* = *Down*). It is worth noting that resumption of interrupted goals differs from recovery of errors (i.e., failed, forgotten or upset tasks). Goals are resumed by inserting a new token in the task model which enters all task templates but bypasses those that have been done in the past (see arc with inscription *s* = *Done*). On the other hand, error recovery is done separately for each task without the need to re-activate the whole group of tasks that comprise a certain goal. Hence, error recovery is modelled by allowing a failed, forgotten or upset task to enter the 'task template' of the specific task via the global-fusion place *Recovered_Tasks* from the decision model of the executive mechanism. Error consequences are considered in the user model which updates accordingly the information held in place *Task_State_List* and the information carried by the variable *impact* to transition *End_Task*. If the impact of an error prevents further processing of the remaining tasks (i.e., *impact* = *Locked*), then the task token is consumed in this transition (i.e., condition *pass* = *on* evaluates to false) and this part of the model reaches an end state.

4.3 The executive mechanism

The executive mechanism basically consists of the monitoring and decision-making transitions and runs in parallel to the task and user models (Fig. 6). A closed

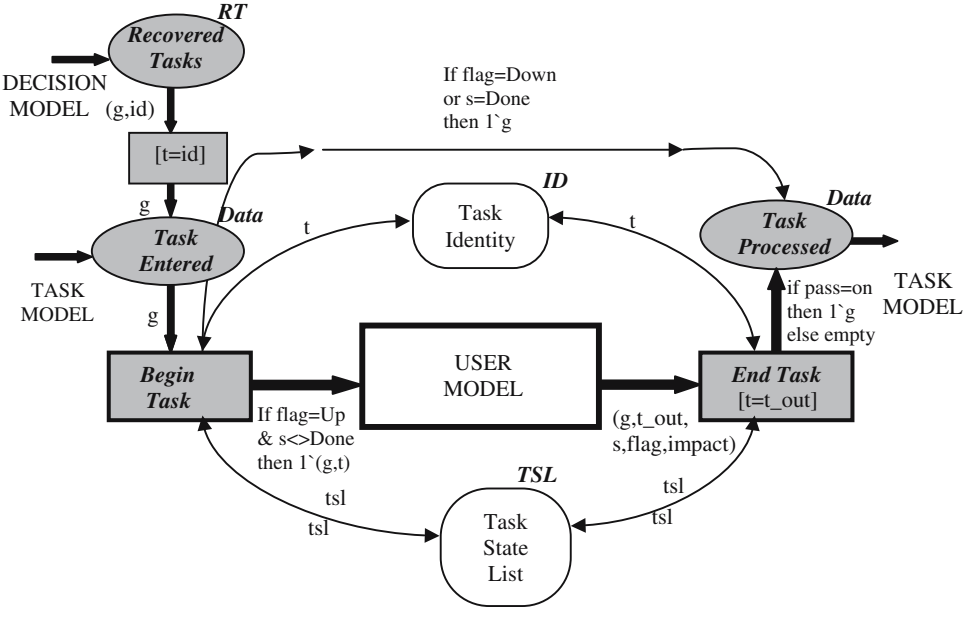

Fig. 5 A task template that communicates data between the task and user models. Error recovery is initiated via the decision model of the executive mechanism

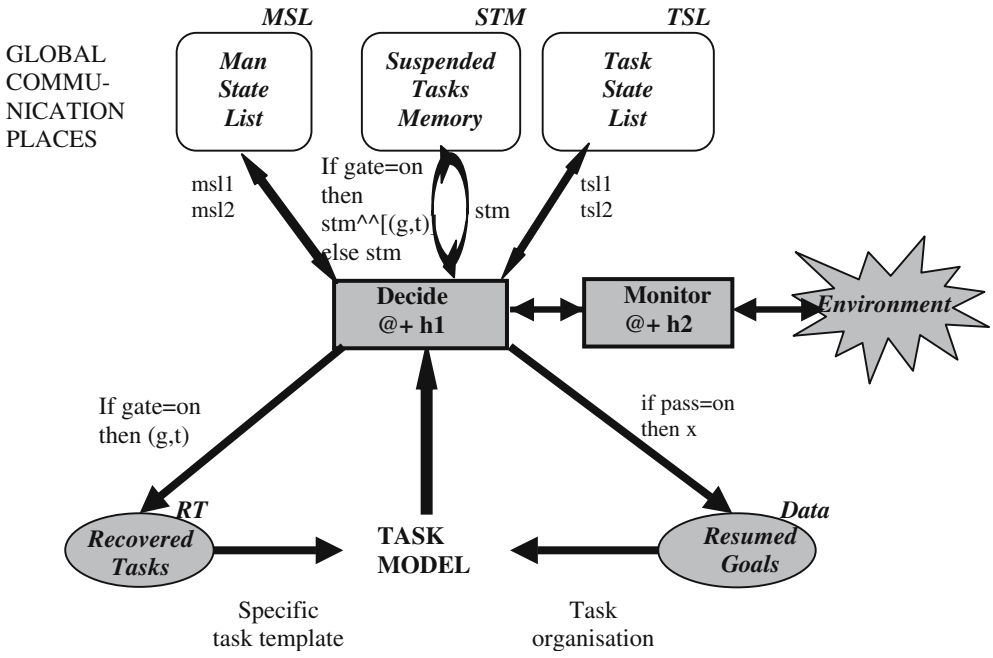

Fig 6 The executive mechanism that monitors the environment and makes decisions about goal processing and error recovery

loop between the two transitions ensures that samples of the work environment are taken and monitored at regular intervals and decisions are re-considered at a later time in transition *Decide*. The stages of monitoring and decision-making are performed in cycles as defined by time intervals *(h1, h2)* respectively. The decisions concern changes in the processing of goals (e.g., interruption or resumption of goals) and potential recovery of failed, forgotten and upset tasks. Figure 6 shows that resumption of goals is done in a different manner from error recovery of tasks.

When a goal is interrupted, the "task token" visits but bypasses all associated tasks and finally exits the task model and is fed back to transition *Decide* of the executive mechanism. A decision is made with regard to the appropriate time for resuming the interrupted goal and a new token is sent to the place *Resumed_Goals* (Fig. 6) and the task model for processing any tasks that remain incomplete due to interruptions. On the other hand, failed, forgotten or upset tasks are detected by evaluating information from the global-fusion places *Man_State_List* and *Task_State_List*. A decision is subsequently made with regard to the appropriate time for recovering these tasks. When colleagues or external cues remind operators of a forgotten task, a token is sent directly to the template of the specific task and information is updated in a global-fusion place *Suspended_Tasks_Memory*. The same procedure is followed for recovering tasks that have failed or have been upset by human errors.

4.4 The cognitive user model

The cognitive user model adapts the task network to the changing work conditions and comprises a set of modules that address attention management, workload management, mental tracking of tasks and human reliability. The cognitive user model (Fig. 7) consists of four transitions that read and write to five global communication places. Each 'task token' enters the user model in place *User_Model_In* and is added to the queuing place *Candidate_Tasks*. In addition, interrupted and deferred tasks are added to the queue of candidate tasks from the place *Suspended_Tasks_Memory*. When memory load exceeds a threshold, some tasks may be forgotten and shed from S*uspended_Tasks_Memory*. Shed tasks should bypass the stages of operator assignment and execution and are put directly at the output of the user model. Transition *Recall_Tasks* keeps mental tracking of all candidate tasks while transition *Prioritize_Tasks* selects tasks according to certain priority criteria.

From the queue of prioritized tasks in place *Selected_Tasks*, transition *Assign_Tasks* selects tasks in order to assign suitable personnel that should execute them. When the assignment of operators to tasks is completed, transition *Assign_Tasks* produces another queue of tasks in place *Active_Tasks* that are subsequently processed in transition *Execute_Tasks* by the assigned personnel. Transition *Assign_Tasks* calculates the workload of operators and implements a workload management strategy (e.g., tasks may be done in series instead of parallel or tasks may be assigned to less busy operators). Human errors may occur during the execution stage in transition *Execute_Tasks* and are communicated to place *Task_State_List*.

4.4.1 Attending tasks in the cognitive user model

Attending tasks involves generating a list of candidate tasks from which certain tasks are selected according to

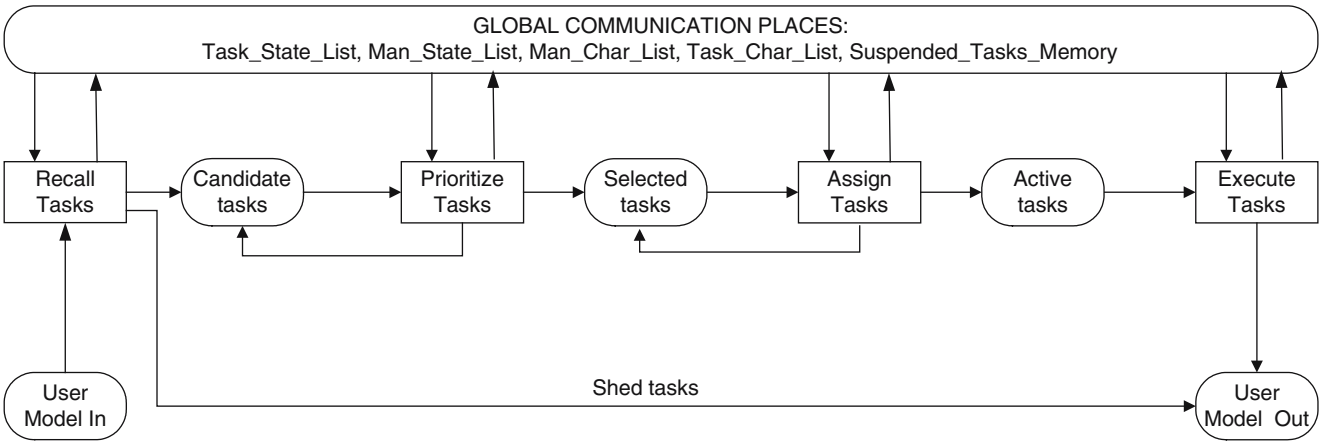

Fig. 7 An overview of the cognitive user model (redescribed in two CPN models in Figs. 8, 9)

pre-defined criteria (see CPN model in Fig. 8). Transition *Recall_Tasks* generates a queue of candidate tasks by adding up current tasks and tasks retrieved from *Suspended_Tasks_Memory* (i.e., interrupted and deferred tasks). The memory activation model (Sect. 3.2) calculates the activation levels of suspended tasks (Eq. 3) and sheds those that are below a threshold. Transition *Recall_Tasks* performs this calculation by opening a file to obtain data about the cost of forgetting and the strength of external cues for each individual task. The effect of workload on mental tracking is taken into account in this calculation by obtaining information about the workload of each operator from place *Man_State_List*. Shed tasks are sent directly to place *User_Model_Out* while their status is updated in a stack of tasks held in place *Task_State_List*. The task token in this module is a pair of two variables *(g,t)* identifying the current goal and task while additional task information is held in place *Task_State_List*. The colour or data type of this place is shown in the Appendix and contains additional information that may be needed in other CPN models.

Transition *Prioritize_Tasks* prioritizes tasks according to the task constrains, temporal constrains and priority values calculated for each task token (see selective attention model, Eq. 1). Data that are stable over the running of the CPN model are kept in data files containing information about the task constraints and the costs of interrupting tasks. Dynamic data are obtained from places *Task_State_List* and *Man_State_List* in order to calculate a priority index for each candidate task at the current time frame. Tasks that are not selected at the current time frame are fed back to place *Candidate_Tasks* (i.e., condition *pass = on* evaluates to true) while the selected task is forwarded to the next place *Selected_Tasks*.

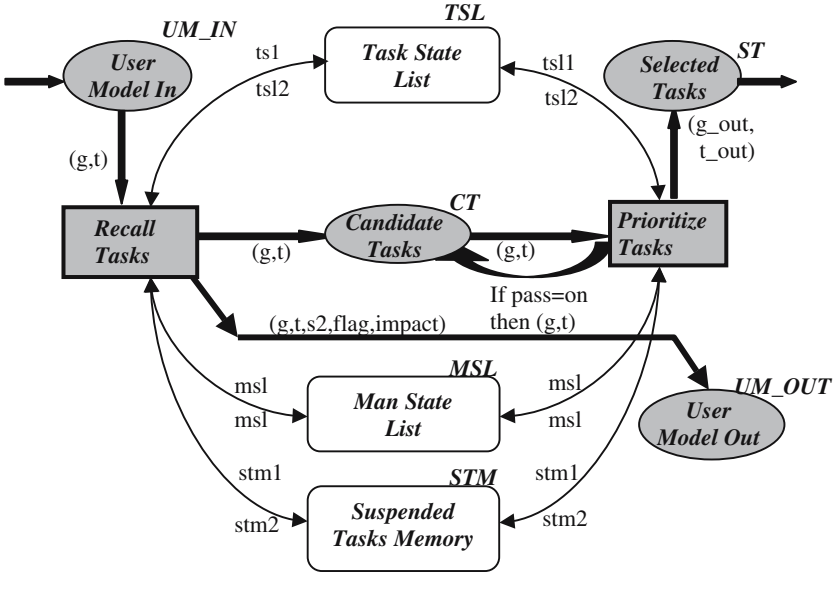

Fig. 8 A CPN model for attending tasks that involves preparing a list of candidate tasks and prioritising tasks

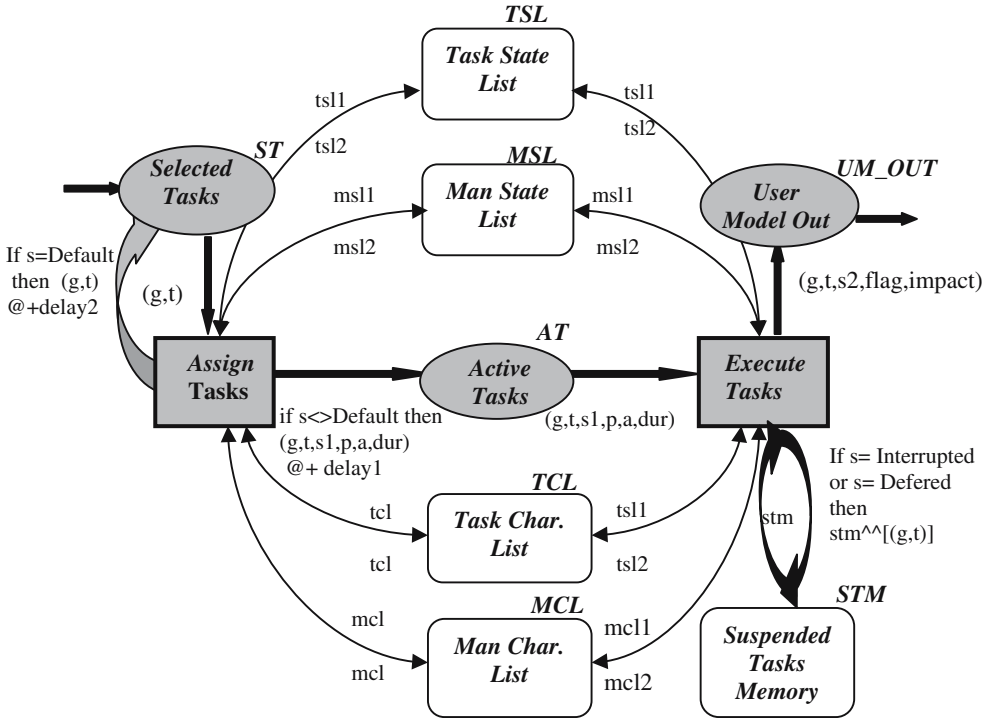

Fig. 9 A CPN model for performing tasks that involves assigning roles, managing workload and executing tasks

4.4.2 Performing tasks in the cognitive user model

Performing tasks involves generating a list of active tasks which are executed according to their time stamp (see CPN model in Fig. 9). Transition *Assign_Tasks* uses the algorithm described in Sect. 3.3 in order to assign tasks to operators according to Eq. 4. Task data for this algorithm come from global-fusion place *Man_Char_List* which logs suitable personnel for each task, and their capabilities in terms of accuracy and speed (or task duration). Therefore, a list of personnel is produced and the one with the largest assignment score is selected to do the current task. In order to take into account the workload level of operators, transition *Assign_Tasks* computes a workload index. When the workload of the selected operators exceeds a threshold, it is possible to choose from the workload management strategies in Sect. 3.1. For instance, if a serial processing strategy is chosen then the task token would be routed back to place *Selected_Tasks*. To estimate the workload index, task data are used from global-fusion place *Task_Char_List* which keeps a list of records about task loadings on the perceptual, cognitive and motor channels. Prospective workload is calculated for operators who perform other tasks in parallel to the current one; this information about parallel tasks is provided from global-fusion place *Task_State_List*. Finally, the workload levels and availability schedule of operators are updated in global-fusion place *Man_State_List* (see Appendix for the data type or colour of all places).

Tasks that have been assigned to an operator are queued in place *Active_Tasks* with a time stamp (@ + delay) indicating their starting time and duration. When this time elapses, the respective task tokens are executed by transition *Execute_Tasks* which also performs an analysis of human errors and error consequences. Human errors may occur during the execution stage which are communicated to place *Task_State_List*. Regardless of the degree of success, a task token is produced in place *User_Model_Out* to convey information to the task templates and the task network.

Human errors are induced in the simulation via two modes or a combination of both: (1) a *manual insertion mode* where the designer can manually insert different types of errors in order to assess the impact upon task scheduling and work organization and (2) *the error inducing function* (EIF) *mode* where an EIF function is used to calculate the effects of workload, fatigues and mental-tracking load upon human reliability. The EIF algorithm (adapted from Cacciabue et al. 2003) takes into account the values of these indices of performance and their weights (or importance) for each action. If the produced EIF value exceeds a threshold, then an error occurs. Because it is very difficult to relate the EIF to the error types (i.e., omission, mistiming and prolongation errors), these are selected either manually or on the basis of a pre-set probability function.

The user model can produce on-line simulations of omission errors, mistiming, interruptions and delays in the *Execute_Tasks* transition and mental-tracking errors in the *Recall_Tasks* transition. Other erroneous actions identified by Hollnagel (1993), such as reversals, replacements and intrusions, are modelled separately in the task model as they are related to changes in the task sequence.

Error consequences can take the following forms: impact on current task (i.e., accuracy and task duration), impact on assigned operator (i.e., increase in workload or change of personnel), side-effects on completed tasks, and inaccessibility to subsequent tasks until problem is resolved. The error consequences can incur either immediately upon task failure or at a later stage when another attempt is made to repeat this task. When a task is interrupted or deferred, an additional token is sent to place *Suspended_Tasks_Memory*. When an error occurs, information about failed and upset tasks is posted to place T*ask_State_List* to be used by the executive mechanism.

5 Analysis modes of the simulation tool

The proposed tool can be used to simulate crew performance and examine aspects of system performance and human reliability. The tool takes different task demands as inputs, calculates the PSF (i.e., workload, mental tracking load, and fatigue) and produces several metrics of performance (i.e., job duration, job accuracy, number of interrupted or forgotten tasks, errors). The adverse effects of task demands can be reduced by manipulating two mitigating factors, that is, workload management and work organization. The tool affords two types of analysis, namely: (1) state space analysis in order to verify the structure and behaviour of the task network and user model, and (2) simulation analysis in order to generate data about human performance, or validate aspects of the cognitive user model.

5.1 State space analysis

The mathematical foundation of Petri nets provides the basis for using a variety of formal analysis techniques that are available in DesignCPN to verify dynamic properties of the system. It is possible to examine, for instance, whether the task network contains deadlocks, never-ending tasks, dangling tasks that do not contribute anything, and tasks that are activated unintentionally resulting in a lack of synchronization. These properties of liveness, boundness and fairness can be investigated with formal analysis techniques, such as reachability graphs, state equivalent graphs, and sweep-line techniques.

Basically, each state of the CPN model corresponds to a list of all places and a distribution of tokens in the places. A state space consists of a large number of states that may be searched and analysed with formal analysis techniques. In the *manual insertion mode*, the state space of the discrete model of the case study ranged between 2000 and 5000 states depending on the consequences of errors and the cycles of monitoring and decision-making. Duration levels of tasks had discrete values and human errors were inserted manually. This was a relatively small state space with a small number of dead markings (i.e., state-nodes), none of which was unpredictable. There were no dangling or never-ending tasks and no lack of synchronization.

In the EIF *mode*, however, the size of state space was increased by an order of magnitude when different probability values were associated to several error types. It is possible to treat task duration as a stochastic variable, however, this increases exponentially the size of the state space. Equivalent state graphs and the sweep-line technique can be used to reduce the size of the state space but their application was beyond the scope of the study. In recent literature, a growing number of studies are exploring the application of state space analysis to operational planning using Coloured Petri nets (Petrucci et al. 2002; Kristensen et al. 2002).

5.2 Simulation analysis

The simulation tool can model crew performance and examine the interaction between task demands, workload-management strategies and work organization schemes. System performance and human reliability can be improved by modifying these three factors. For example, task demands can be reduced by adjusting the organization of tasks in the task model, by redesigning tasks (i.e., task duration, task priorities, and external cues for suspended tasks) and by removing some consequences of task failures. On the other hand, operators can resort to different strategies for reducing their workload and enhancing their reliability. Finally, work organization schemes can be optimized by changing the task allocation rules, the task range and the degree of expertise of operators. It is also worth noting the interaction between these factors. For instance, the range of viable workload strategies may depend on the task demands (e.g., it may not be possible to defer tasks or perform tasks in a serial fashion) and on the expertise of operators (e.g., higher levels of expertise may reduce workload or affect the choice of workload strategies).

The simulation tool produces several system and psychological metrics to assess the impact of changes in task demands, workload strategies and work organization. System metrics may include accuracy and time to perform the task sequence, state of tasks at any point in time (e.g., interrupted, deferred, shed tasks), the assignment of tasks to operators and time periods when preferred operators were unavailable. Psychological metrics may include indices of workload, fatigue and mental tracking load.

6 Application to a case study in process control

To demonstrate the interaction between task demands, workload management strategies and work organiza-

tion, two cases have been developed: Simulation Run 1 and Simulation Run 2. Two work organization schemes were evaluated in terms of their impact on system performance and human reliability, namely: (1) a serialist work organization where the supervisor assumes responsibility for the majority of tasks and (2) a universalist work organization where both operators are capable of performing most tasks. In general, a serialist organization achieves higher standards in task accuracy because of the higher specialization of operators in the tasks they are trained to perform. On the other hand, a universalist organization can apply a wider range of workload management strategies because less busy operators can undertake a wider range of tasks.

The objective of the case study was to compare the two work organization schemes in terms of their potential to mitigate the adverse effects of task demands on system performance.

The case study refers to a process control task comprising fifteen tasks to be performed by a supervisor and an assistant. The normal job included eleven tasks grouped into two goals and four contingent tasks grouped into a third goal to be carried out depending on a set of emergency conditions. The main part of the job involves deciding how to allocate tasks between the two operators so that goals 1 and 2 are carried out in parallel for most of the time. The task network (Fig. 10) shows the pre-conditions for initiating tasks (e.g., task 3 should start when tasks 1 and 2 are completed). The allocation of tasks to operators is performed in transition *Assign_Tasks* (Fig. 9) which takes into account a set of global and local allocation rules, the skills of operators in terms of speed and accuracy, and the workload of operators. The supervisor is responsible for monitoring the plant conditions on a control panel and makes decisions on how to cope with emergent events.

In the current scenario, an emergent event requires a prompt response by initiating recovery goal 3 and temporarily suspending goal 1 or goal 2. The suspended goal 1 or 2 can be resumed, once tasks 12 and 13 of goal 3 (Fig. 10) have been performed successfully. Monitoring the control panel and making decisions on how to cope with emergent events or human errors are activities assigned to the supervisor, which can increase his or her workload. For the purpose of the case study, performance decrements and errors have been determined mainly by operator workload.

7 Simulation Run 1: A serialist work organization

In the serialist organization, there is a high degree of specialization where the assistant is capable of performing only half of the required tasks (i.e., tasks 3, 5, 6, 9, 11, 14, 15) but to a higher standard of task accuracy than the supervisor who can perform all tasks. The operator with the higher assignment score is first assigned to each task but this policy occasionally creates

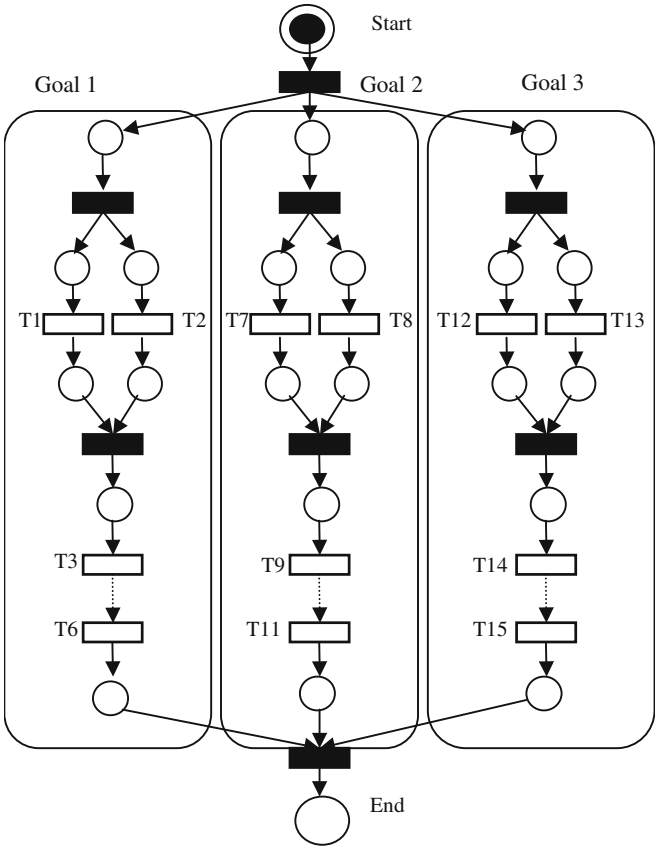

Fig. 10 The task model of the case study comprising 15 tasks organized into goal 1, goal 2 and the recovery goal 3

high workload levels. Because of the serialist organization, however, it is not always possible to reassign tasks to the other operator due to lack of expertise. Potential workload management strategies include: resorting to serial rather than parallel processing, deferring the new task, and interrupting ongoing tasks in favour of new

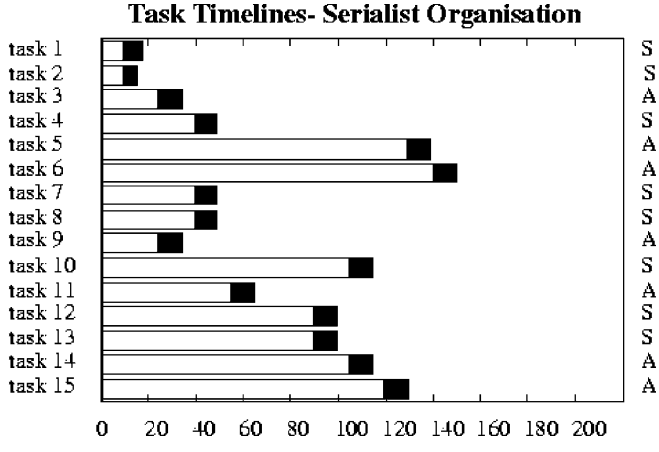

Fig. 11 A task timeline in the serialist organization showing task allocation to the supervisor (*S*) and the assistant (*A*)

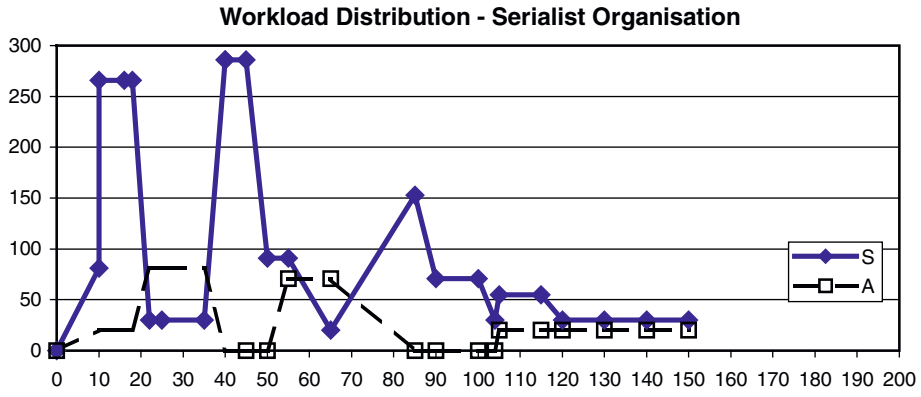

Fig. 12 A trace of the workload of the supervisor (*S*) and the assistant (*A*) in the serialist organization (workload values above 150 may result in errors)

tasks. When task demands preclude the application of such strategies, human performance may deteriorate resulting in lower standards of accuracy or human errors.

Simulation Run 1 produces a task timeline with the assigned operators (Fig. 11) and a workload trace of operators (Fig. 12). The workload of the supervisor becomes very high in the interval 10–20 as s/he has to perform tasks 1, 2, 7 and 8 in parallel. The reason is that it is not possible to allocate any of these tasks to the assistant (i.e., lack of expertise) nor it is possible to defer new tasks or interrupt ongoing tasks (i.e., high task demands). The high workload results in an error, that is, the supervisor fails to perform task 8 correctly. Hence, s/he has to repeat both tasks 7 and 8 at some time in future since the two tasks should be executed in parallel. When this time comes (40–50), tasks 7 and 8 are executed correctly but coincide with performance of tasks 4 and 10. Supervisor workload increases again and the only viable strategy is to defer either task 4 or task 10. At this point, the scenario simulated the decision of the supervisor to perform all tasks in parallel which resulted in failing to perform task 10 correctly. This is a worst case scenario in comparison to a scenario based on deferring tasks 4 or 10.

A few minutes later (65), the supervisor detects an abnormal event and orders that the recovery goal 3 is initiated. Because only two goals can be processed at a time, goal 1 is suspended while goal 2 is revisited to execute task 10. Parallel processing of tasks 10, 12 and 13 leads to high workload (at time interval 85) and the supervisor decides to defer task 10 in favour of recovery goal 3 (Fig. 12). When task 10 is performed correctly, the operators resume goal 1 and perform the remaining tasks 5 and 6. In the recovery stage of the scenario, the workload of the supervisor is always below the threshold value (150) since the assistant is competent in performing tasks 5, 6, 14 and 15.

7.1 Simulation Run 2: A universalist work organization

In the universalist organization, there is a low degree of specialization where both operators can perform all tasks but at a lower level of task accuracy than the serialist organization. The operator with the higher assignment score is first assigned to the task but this policy is modified when the most competent operator has a high workload level. In this sense, workload is kept under the specified threshold value (150).

Simulation Run 2 produces a task timeline with the assigned operators (Fig. 13) and a workload trace of operators (Fig. 14). The workload of the supervisor becomes quite high in the interval 10–20 as s/he has to decide whether to perform tasks 1, 2, 7 and 8 in parallel or not. To reduce the high workload, the supervisor assigns tasks 1 and 2 to the assistant who executes them in parallel as specified in the task network. Tasks 3 and 9 are assigned to the assistant while tasks 4 and 10 are assigned to the supervisor according to their assignment scores. Goal 2 is completed when the assistant performs task 10.

A few minutes later (65), the supervisor detects an abnormal event and orders that the recovery goal 3 is initiated. Because only two goals can be processed at a time, goal 1 is suspended since goal 2 has already been completed. When tasks 12 and 13 of the recovery goal 3

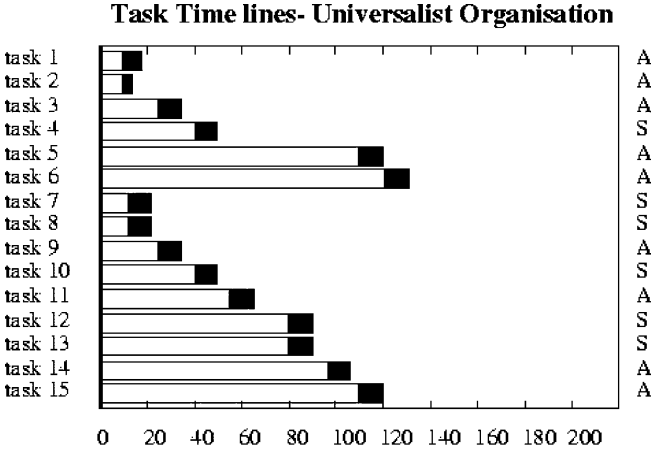

Fig. 13 A task timeline in the universalist organization showing task allocation to the supervisor (*S*) and the assistant (*A*)

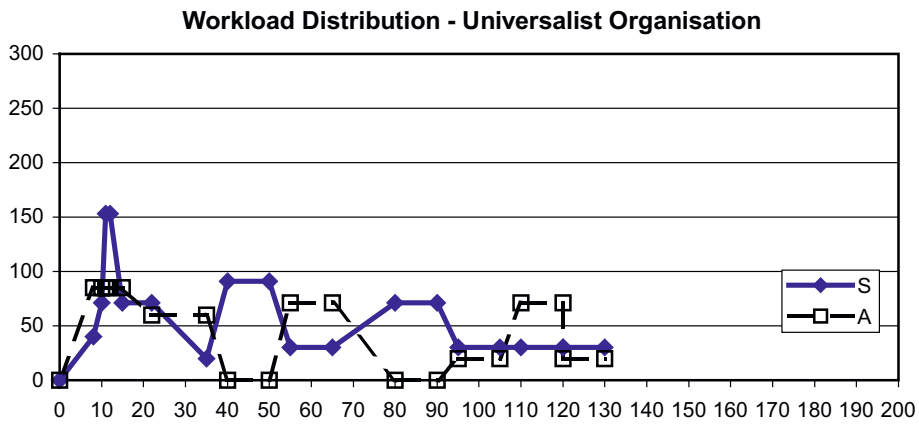

Fig. 14 A trace of the workload of the supervisor (*S*) and the assistant (*A*) in the universalist organization

have been performed, the operators resume goal 1 and perform the remaining tasks 5 and 6. In Simulation Run 2, the workload of both operators was always kept below the threshold since they were both capable of performing all tasks.

7.2 Comparison of simulation runs

The results of the case study show that the universalist organization is a more flexible work scheme for this application because it can deal better with periods of high workload than the serialist one. The high workload of the supervisor in the serialist organization gave rise to two human errors that were recovered successfully at later periods of time. As a result, the total task duration in the serialist organization ($t = 150$) was longer than the universalist one ($t = 130$). However, these benefits of the universalist organization have to be traded-off against task accuracy. The wider task repertoire of operators in the universalist organization implies fewer opportunities for practice and lower accuracy scores than the ones found in the more specialized organization. The analysis showed that the accuracy score in the universalist organization was approximately 20% lower than the one reported in the serialist organization. The case study illustrates that workload management strategies and work organization are two important mediating factors in the impact of task demands upon human performance. An assumption was made that operators in the universalist organization could perform individual tasks without errors (i.e., lack of practice would affect task accuracy only) and that they would be able to use a range of workload management strategies. Different results could be obtained by experimenting with different assumptions about these mediating factors (e.g., Zulch et al. 2003).

The results also indicated that the most critical task demand was the requirement to perform tasks 1 and 2 in parallel to tasks 7 and 8. In the serialist organization, this resulted in high workload because the supervisor could not defer or perform them in serial and neither could s/he assign some of them to the assistant who was specialized in other tasks. Some general recommendations for improving human reliability in the case study may include:

(i) Reduce task demands by redesigning the job to minimize the need for parallel processing or minimize the consequences of task failures,
(ii) Reduce workload in parallel processing of tasks 1, 2, 7 and 8 by automating parts of the job, simplifying the job or training operators to over-learn responses, and
(iii) Change work organization so that the assistant acquires a wider repertoire of skills.

It is worth noting that all recommendations regard human factor interventions in transitions *Assign_Tasks* and *Execute_Tasks* (Fig. 9). Interventions in the cueing features of tasks, priorities of tasks and consequences of suspending tasks (i.e., transitions *Recall_Tasks* and *Prioritize_Tasks*, Fig. 8) were not applicable in the present case study.

8 Conclusions

The simulation tool has been developed to serve the requirement of integrating task networks with user cognitive models. Task networks can generate several performance metrics but cannot cope with variations in job design, such as modifications to job procedures, goal priorities and task allocation. Integrating task networks with cognitive models allows designers to use several system and psychological metrics (i.e., duration and accuracy scores, indices of workload and fatigue) as well as model how crew performance can be adapted to changing circumstances (i.e., mental tracking of suspended tasks, coping with interruption and reallocation of tasks, and managing workload). Hence, adapting to job variations and recovering from early unsuccessful attempts becomes the basis of job design for safety and productivity.

An important feature of the simulation tool is that human performance is examined within the wider context of work. For instance, the performance of a task is also affected by the state of other operational tasks (i.e., candidate, suspended and failed tasks) and the state of mental activities of operators (i.e., monitoring events, mental tracking, dividing attention and managing workload). As a result, specific recommendations can be derived for improving productivity and reliability by means of changes in task demands, workload strategies and work organization. The simulation tool is currently under development in order to address three broader issues: - the validity of cognitive models, the mechanisms of user models that affect human reliability, and the validity of information used in task modelling.

In order to obtain valid results, user models should be valid models of the interaction between cognition and work. In this sense, the models of selective attention and work organization are preliminary models that have not been validated in the context of real task scenarios. On the other hand, the models of divided attention and memory management have been validated in the literature (Wickens et al. 1988; Altman and Trafton 2002) in diverse applications (i.e., air traffic control, aircraft control, and human computer interaction) but it is difficult to ascertain the extent that the findings can transfer to process control tasks. Empirical studies are, therefore, needed to validate the utilized user models in the context of process control. However, it is anticipated that simulation is, by itself, a useful tool in comparing simulated user models to empirical data. Future developments of the cognitive user model of the simulation tool should proceed in this direction.

Another issue concerns the mechanisms of the cognitive user model that influence human reliability. At the present stage of development, the simulation tool assumes that human errors arise as a result of a set of PSF (e.g., workload, fatigue, mental tracking load) collectively exceeding a threshold. However, human performance is flexible and operators may avoid an error by reducing the requirements of performance (e.g., perform a task to a lower degree of accuracy). Therefore, it is difficult to determine whether high PSF levels will give rise to human errors or performance decrements. In addition, it is not possible to determine the types of error that may be the result of high task demands. The simulation tool should be mainly used to compare different task demands and identify worst case scenarios that affect system performance rather than as a representation of cognitive processes along the lines of Yoshikawa et al. (1997).

The issue of data and information of the user models or algorithms is another major concern. At present, the simulation uses rather arbitrary values for many of the variables regarding task attributes (e.g., temporal and sequence attributes), ratings on the information processing channels, external cues associated with suspended tasks, error consequences, deadlines associated with competing tasks and so on. These data are specific to the nature of the application tasks and should be acquired by means of field observation and experimentation. A number of empirical studies are required in order to collect data that can be used to calibrate the variables and thresholds of the algorithms of the cognitive user model. This is necessary in order to use the simulation to make predictions about human performance in new tasks.

When these validity issues are resolved, the simulation tool can be used to redesign jobs and teams in order to increase productivity and reliability. Designers may be able to verify operating procedures for a wide range of job variations and identify critical task demands that could give rise to human errors. Job design can also benefit from the simulation tool by redesigning task features to reduce workload, introducing additional cues for tasks, and changing adverse consequences arising form suspended or failed tasks. Finally, strategies for managing workload, changing goal priorities and rescheduling tasks can form the basis for new training schemes.

From a computer science viewpoint, CPN and its Standard ML language were a useful modelling platform for procedural tasks. DesignCPN, however, may run into difficulties when modelling reactive systems in which monitoring certain events takes priority over others. The lack of ''priority'' transitions makes modelling of certain tasks rather cumbersome (e.g., monitoring events and performing tasks at the same time). In addition, the modelling of interrupting events is inadequate since tasks or transitions cannot be literally interrupted once execution has started. Interrupting a task, which has already been on progress, would require a change of the time model of DesignCPN towards ''augmented time'' models advocated by Zhou and Venkatesh (1998). Further improvements in the architecture of Petri nets would provide additional features for modelling the temporal dynamics of complex systems.

There is little doubt that computational modelling efforts may require extensive resources in terms of data, expertise and time. However, modern systems require an increasing degree of flexibility in scheduling and adaptation of job procedures to the changing work conditions. The proposed simulation tool offers a platform for modelling system performance and human reliability with a similar modelling language that is amenable to both simulation and formal analysis.

9 Appendix

Selected declaration of colours or data types used in the CPN models

```
colour INT = integer;
colour Delay = INT;    colour  Data, ID  = INT;    colour  Goal, Task = INT;
colour Accuracy, Duration  = INT;
colour Person = with None | All |  Supervisor |   Assistant ;
colour PAD =  product Person * Accuracy * Duration ;
colour Flag = with Up | Down;  (* when goal is interrupted, the flag is down*)
colour Impact = with Locked | Unlocked;  (* side-effects of task failures *)
colour Pass, Gate = with on | off;
colour State = with  Default     | Started        | Done         | Deferred
                     | Upset      | Shed          | Failed       | Interrupted;
colour UM_IN =  product Goal * Task,
colour UM_OUT= product Goal * Task * State * Flag * Impact ;
colour CT, ST, RT  = product Goal * Task;
colour AT = product Goal * Task * State * Person * Accuracy * Duration ;

colour ManChar = record
      tid:INT *                (* task identification number *)
      eff: list PAD;           (* list of  tuples of type PAD *)

colour TaskChar = record
      gid:INT*                 (* goal identification number *)
      tid:INT*                 (* task identification number *)
      P:INT*                   (* loading on perception channel *)
      C:INT*                   (* loading on cognitive channel *)
      M:INT;                   (* loading on motor channel*)

colour ManState = record
      name:Person *            (* name of person participating in the job*)
      avahr:INT *              (* time that this person  becomes available again*)
      work:INT;                (* current level of workload)

colour TaskState =   record
      tid:INT  *               (* task identification number *)
      pri:INT  *               (* default priority number of task *)
      flag : Flag *            (* status of  associated goal Up or Down *)
      person:Person *          (* person assigned to perform the task *)
      status:State *           (* status of task at current time *)
      tb:INT *                 (* time that task started *)
      te:INT ;                 (* time that task ended *)

colour STM = list ST;
colour TSL  = list TaskState ;   colour MSL = list ManState;
colour MCL = list ManChar;    colour TCL = list TaskChar ;

(* Global Fusion places:  Suspended_Tasks_Memory, Recovered_Tasks, UM_IN, UM_OUT,
Task_State_List, Man_State_List, Man_Char_List, Task_Char_List    *)
```

References

van der Aalst WMP, Hofstede AHM (2002) Workflow patterns: on the expressive power of (Petri nets based) workflow languages. In: Jensen K (eds) Proceedings of the fourth workshop on the practical use of coloured Petri nets and CPN tools, Department of Computer Science, University of Aarhus, Denmark, vol 560, pp1–20

Abed M, Tabary D, Kolski C (2003) Using formal specification techniques for modelling tasks and generation of HCI specifications. In: Diaper D, Stanton N (eds) The handbook of task

analysis for human computer interaction Chapter 30. Lawrence Erlbaum Associates, Mahwah, pp. 503–529
Altman EM, Trafton JG (2002) Memory for goals: an activation-based model. Cogn Sci 26:39–83
Blom HAP, Daams J, Nijhuis HB (2000) Human cognition modelling in ATM safety assessment. In: Proceedings of the 3rd USA/Europe AirTraffic Management R & D Seminar, Napoli, 13–16 June 2000
Cacciabue PC (1998) Modelling and simulation of human behaviour in system analysis. Springer, Berlin Heidelberg New York
Cacciabue PC, Mauri C, Owen D (2003) The development of a model and simulation of an aviation maintenance technician performance. Cogn Technol Work 5:229–247
Card SK, Moran TP, Newell A (1983) The psychology of human computer interaction. Lawrence Erlbaum Associates, Mahwah
DesignCPN version 4.0, Meta Software Corporation. Cambridge MA. Also available from the Department of Computer Science, University of Aarhus at http://www.daimi.au.dk/DesignCPN
Embrey DE, Kontogiannis T, Green M (1994) Guidelines for preventing human error in process safety Center for Chemical process safety. American Institute of Chemical Engineers, New York
Freed M (1998) Managing multiple tasks in complex, dynamic environments. In: Proceedings of the 15th national conference on artificial intelligence, AAAI Press/MIT Press, Madison/Wiscosin. pp. 921–927
Freed M (2000) Reactive prioritization. In: Proceedings of the 2nd NASA international workshop on planning and scheduling in space, March 2000. San Francisco
Handley HAH, Levis AH (2003) Modelling for future command and control architectures. In: Proceedings of the 8th international command and control research and technology symposium, National Defense University, Washington DC, 17–19 June 2003
Handley HAH, Zaidi ZR, Levis A (1999) The use of simulation models in model-driven experiments. Syst Eng 2(2):108–128
Hendy KC, Farrell PS (1997) Implementing a model of information processing in a task network simulation environment (DCIEM 97-R-71). Defense and Civil Institute of Environmental Medicine, Ontario, Canada
Hollnagel E (1993). Human reliability: context and control. Academic, London
Jensen K (1997a) Coloured Petri nets: basic concepts, analysis methods and practical use. vol 1: basic concepts. Monographs in theoretical computer science, Springer, Berlin Heidelberg New York
Jensen K (1997b) Coloured Petri nets: basic concepts, analysis methods and practical use. Vol 2: analysis methods. Monographs in theoretical computer science, Springer, Berlin Heidelberg New York
John BE, Kieras DE (1996) The GOMS family of user interfaces analysis techniques: comparison and constructs. ACM Trans Comput Hum Interact 3(4):20–351
Kirwan B, Ainsworth LK (1992) A guide to task analysis. Taylor & Francis, London
Kjellen U (2000) Prevention of accidents through experience feedback. Taylor & Francis, London
Kletz TA (1991) An engineer's view of human error, 2nd edn. Institution of Chemical Engineers, Rugby
Kontogiannis T (2003) A Petri-net based approach for ergonomic task analysis and modelling with emphasis on adaptation to system changes. Saf Sci 41:803–835
Kristensen LM, Mitchell B, Zhang L, Billington J (2002) Modelling and initial analysis of operational planning processes using coloured Petri nets. In: Proceedings of workshop on formal methods applied to defence systems, conference in research and practice in information technology, Australian Computer Society, vol 12, pp105–114
Lacaze X, Palanque P, Bastide R (2002) Performance evaluations as a tool for quantitative assessment of complexity of interactive systems. Lecture Notes in Computer Science, Springer Verlag, No 2545, pp 208–222
Laughery KR, Corker KM (1997) Computer modelling and simulation of human-system performance. In: Salvendy G (ed) Handbook of human factors and ergonomics, 2nd edn. Wiley, New York, pp 1375–1408
Lebiere C, Biefeld F, Archer S, Allender L, Kelley TD (2002) IMPRINT/ACT-R: integration of a task network architecture with a cognitive architecture and its application to human error modelling. In: Chini MJ (ed) Military, government and aerospace simulation, Society for Modelling and Simulation International, San Diego 34(3):13–19
Levenson N, Stolzy (1987) Safety analysis using Petri nets. IEEE Trans Software Eng 13:386–397
Lindsay P, Connelly S (2002) Modelling erroneous operator behaviours for an air traffic control tasks. In: Grundy J, Calder P (eds) Third Australian user interface conference, Australian Computer Society Inc, vol 7, pp 43–54
Lockett JF (1997) Task network modelling of human workload coping strategies. In: Smith MJ, Salvendy G, Koubek RJ (eds) Design of computing systems: social and ergonomic considerations, proceedings of the seventh international conference on human–computer interaction, (HCI International '97). Elsevier, San Francisco vol 2, pp 71–74
Lovett MC, Daily LZ, Reder LM (2000) A source activation theory of working memory: cross task prediction of performance in ACT-R. J Cogn Syst Res 1:99–118
Moher T, Dirda V (1995) Revising mental models to accommodate expectation failures in human–computer dialogues.In: 2nd DSVIS workshop on design, specification and verification of interactive systems, Speringer, Berlin Heidelberg New York
Mori G, Paterno F, Santoro C (2002) CTTE: support for developing and analyzing task models for interactive system design. IEEE Trans Software Eng 28(9):1–17
North RA, Riley VA (1989) A predictive model of operator workload. In: McMillan GR, et al (eds) Applications of human performance models to system design. Plenum Press, New York pp 81–90
Palanque P, Bastide R (1997) Synergistic modelling of tasks, users and systems using formal specification techniques. Interact Comput 9(2):129–153
Petrucci L, Kristensen LM, Billington J, Qureshi Z (2002) Towards formal specification and analysis of avionics mission systems. In: Proceedings of conference in research and practice in information technology, Workshop on formal methods applied to defense systems, Australian Computer Society, vol 2, pp 95–104
Pew RW, Mavor AS (1998) Modelling human and organizational behaviour: application to military simulations. National Academy Press, Washington, DC
Rauterberg M (1993) AMME: an automatic mental model evaluation to analyze user behaviour traced in a finite, discrete state space. Ergonomics 36:1369–1380
Shepherd A (2001) Hierarchical task analysis. Taylor & Francis, London
Schlick C, Reuth R, Luczak H (2002) A comparative simulation study of work processes in autonomous production cells. Hum Factors Ergon Manuf 12(1):31–54
Stanton N, Young MS (1999) A guide to methodology in ergonomics. Taylor & Francis, London
Stutz P, Onken R (2001) Adaptive pilot modelling for cockpit crew assistance: concept, realization and results. In: Proceedings of the 8th conference on cognitive science approaches to process control. The cognitive work process: automation and interaction. Munich, 24–26 September 2001
Tarby JC, Barthet MF (1996) The DIANE+ method. In: Vanderdonckt J (ed) Proceedings of the 2nd international workshop on computer aided design of user interfaces (CADUI' 96). Presses Universitaires de Namur, Narmur, vol 15, pp 95–120
Uhr H (2003) TOMBOLA: simulation and user-specific presentation of executable task models. In: Jacko J, Staphanidis C (eds) Human computer interaction: theory and practice (Part I). Lawrence Erlbaum Associates, Mahwah, pp 263–267

Wickens CD, Harwood K, Segal L, Tkalcevic I, Sherman B (1988) TASKILLAN: A simulation to predict the validity of multiple resource models of aviation workload. In: Proceedings of the 32nd annual meeting of the human factors society. Human Factors Society, Santa Monica, pp 168–172

Yoshikawa H, Nakagawa T, Nakatani Y, Furutra T, Hasegawa A (1997) Development of an analysis support system for man-machine system design information. Reliability Eng Syst Saf 5(3):417–425

Zachary W, Campbell G, Laughery R, Glenn F (1998) Application of human modelling technology to the design, operation and evaluation of complex systems Technical Memo No 980727.9705. CHI Systems, Inc., Lower Gwynedd, PA

Zhou M, Venkatesh K (1998) Modelling, simulation and control of flexible manufacturing systems: a Petri net approach. World Scientific, New Jersey

Zulch G, Kruger J, Schindele H, Rottinger S (2003) Simulation-aided planning of quality-oriented personnel structures in production systems. Appl Ergon 34:293–301