Developing a Model of Cognitive Lockup for User Interface Engineering

Tina Mioch (Tina.Mioch@tno.nl)[1]
Rosemarijn Looije (Rosemarijn.Looije@tno.nl)[1]
Mark Neerincx (Mark.Neerincx@tno.nl)[1,2]

[1]TNO Human Factors, P.O. Box 23
3769 ZG Soesterberg, The Netherlands

[2]Delft University of Technology Man–Machine Interaction Group,
Mekelweg 4, 2628 CD, Delft, The Netherlands

Abstract

This paper presents the development of a cognitive model of cognitive lockup: the tendency of humans to deal with disturbances sequentially, possibly overseeing crucial data from unattended resources so that serious task failures can appear—e.g., in a cockpit or control centre. The proposed model should support the design and evaluation of user interfaces that prevent such failures, being used outside the academic community. Based on the practical cognitive task load theory of Neerincx (2003), this model distinguishes time pressure and number of tasks-to-do as two factors that increase task switch costs and the corresponding risk of cognitive lock-up. The CASCaS architecture proved to fit best with the requirements to incorporate these factors and to support the UI engineering process.

Keywords: cognitive lockup; cognitive modeling; cognitive task load model; cognitive architectures; user interface engineering.

Introduction

Aircraft pilots are faced with a complex traffic environment. Cockpit automation and support systems help to reduce this complexity. Currently, a lot of research is done to improve the onboard management of flight trajectories and the negotiation of trajectory changes with Air Traffic Control. During the flight, many factors may induce changes to the original flight plan, e.g. bad weather, traffic conflicts, or runway changes. Safe operation of aircrafts is based on normative flight procedures (standard operating procedures) and rules of good airmanship, which we will refer to as normative activities. We define pilot errors as deviations from normative activities.

In the past, several cognitive explanations and theories have been proposed to understand why pilots deviate from normative activities (e.g. Dekker (2003)). The European project HUMAN, in which the research described in this paper is done, strives to pave a way of making this knowledge readily available to designers of new cockpit systems. We intend to achieve this by means of a valid executable flight crew model which incorporates cognitive error-producing mechanisms leading to deviations from normative activities. The model interacts with models of cockpit systems in a virtual simulation environment to predict deviations and its potential consequences on the safety of flight. The ultimate objective of HUMAN is to apply this model to analyze human errors and support error prediction in ways that are usable and practical for human-centered design of systems operating in complex cockpit environments.

At the initial stage of HUMAN we performed questionnaire interviews with pilots and human factor experts based on a literature survey of error-producing mechanisms. We identified cognitive lockup to be among the most relevant mechanisms for modern and future cockpit human machine interfaces. We take the definition of cognitive lockup from Moray and Rotenberg (1989) who define the term ‘cognitive lockup’ as the tendency of operators to deal with disturbances sequentially. This has as a result that operators focus on a subpart of a system and ignore the rest of it (Meij, 2004).

In this paper, we discuss factors that can cause cognitive lockup and an architecture of a cognitive model that can be used to help prevent lockup failures during User Interface engineering.

Cognitive Lockup

Previous Research

As the definition from Moray and Rotenberg (1989) shows, cognitive lockup does not occur when people can perform all their tasks consecutively. Therefore they designed a task where this was not possible. Participants were asked to supervise a simulated thermal hydraulic system that consisted of four subsystems. In one scenario they needed only to focus on one fault in one of the subsystems. In another scenario a first fault was followed by a second fault in a different subsystem, which occurred before the participant could have handled the first fault. It was shown that participants shifted attention much later to the second fault then they did to the first fault. Moray and Rotenberg attributed this to limited information processing capacities. In another study that demonstrated cognitive lockup (Kerstholt et al, 1996), participants had to supervise four dynamic subsystems and deal with disturbances. The system included the option to stabilize a subsystem in which additional faults occurred, with which participants acknowledged their understanding of the development of a disturbance over time. Most participants did not use this option and handled the disturbances sequentially.

Cognitive lockup as a phenomenon is related to the rise of automation, but the tendency to proceed with the current task is not new. Meij (2004) investigated cognitive lockup in relation to planning, task-switching and decision making. He found that both prior investments into a task as the time that is needed to complete the task increases the probability of cognitive lockup. No support was found for refrainment of monitoring (a second fire was detected, but not tended to before the first fire was solved), too optimistic scenarios, and lack of resources (the complexity of the first task did not influence the degree of cognitive lockup).

Cognitive Task Load Model

A model that specifies core aspects of cognitive lockup is the cognitive task load (CTL) model of Neerincx (2003). The development of this model is driven by the need for limited and practical theories and models on human cognition to take validation of the theories and models out the laboratory and into the real world, where the environment is more dynamic.

The CTL-model describes load in terms of three behavioral factors: time pressure, level of information processing and number of task set switches (see Figure 1).

Time Pressure The time pressure is dependent on the scenario and the actions of tasks. The scenario provides information on the number of tasks due to events and the actions that are called upon by the tasks can take a long or a short time to handle. A standard measure for the time pressure is:

$$\text{Time pressure} = \frac{\text{time required for tasks}}{\text{time available for tasks}}$$

Humans reach overload when the time pressure is more than 70-80% (Beevis et al., 1994).

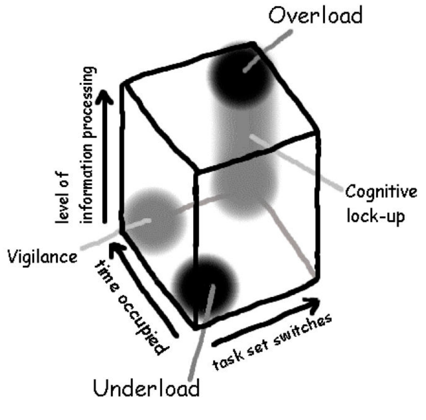

Figure 1: CTL model, with the three dimensions task set switches, level of information processing, and time occupied (time pressure).

Level of Information Processing The level of information processing factor is measured as the percentage of knowledge-based actions using the Skill-Rule-Knowledge framework from Rasmussen (1986). Input information that can be processed at skill level (e.g. when you touch something hot with your hand, you immediately react by removing your hand from the heat source) is not cognitively demanding. When input information triggers a routine consisting of rules (i.e. procedures with rules of the type "if <*event/state*> then <*actions*>") it takes some cognitive capacities to resolve the if/then, but the rest of the procedure is quite automatic. Cognitive demanding are the situations where there is problem analysis needed on the input information and knowledge to reason about it, this can have a large influence on the working memory.

Rasmussen's framework corresponds to the cognitive theory of skill acquisition of Anderson (1982) that distinguishes three memory representations: cognitive, associative and autonomous. These three levels are linked to different memory representations; declarative, procedural and implicit.

Task Set Switches To take into account situations where people have to perform different tasks that appeal to different sources of human knowledge and different objects in the environment, the CTL-model comprises the task set switches factor. A task set contains both the human resources and environmental objects with momentary states, which are involved in the task performance. A switch occurs when the applicable task knowledge on the operating and environment level change. A task set can thus be seen as a goal that is comprised of several (sub-)tasks.

Rubinstein, Meyer and Evans (2001) distinguish two types of task switching: task switching in successive tasks and task switching in concurrent tasks. With successive tasks the first task is responded to and finished before the second task is presented. Concurrent tasks on the other hand are tasks where the second task is presented before the first task has been finished. We are only interested in concurrent tasks, because a pilot usually has multiple concurrent tasks that can be executed, e.g. monitoring different interfaces in the cockpit. Successive task switching studies show that task switching takes time (Jersild, 1927, Rogers & Monsell, 1995). In concurrent task switching studies (De Jong, 1995; Schumacher et al., 1999), it is observed that people are unable to deal with multiple tasks. They postpone the second task until the first task is completed. In these experiments the second task is not of such importance that it should be handled immediately, but in real life situations not handling the second task before finishing the first can cause life threatening situations (e.g. the crash of flight 401 of Eastern Air Lines in 1972 (NTSB, 1973)). Tasks can be interrupted, but with every switch time and effort is needed to do context acquisition to bring the environment information up-to-date (Olsen & Goodrich, 2003).

In the CTL-model, the task set switches can be seen as the number of task set switches possible at a particular moment in time. This number comes thus forth from the environment and the situation a person is in.

Cognitive Lockup in the CTL Model The three factors of the CTL model are interrelated (Figure 1). Cognitive lockup is independent of information processing level, but does occur when both time pressure and number of task set switches is high. That the information of processing level is not of importance seems counterintuitive, but in an experiment of Meij (2004) (experiment 2) this is supported. In the experiment of Meij, participants were asked to monitor for fires on a ship. When a fire was detected it had to be diagnosed on both priority and treatment. Two fires could exist simultaneously and the participant had to decide which fire to fight. The complexity of this task was varied by making the diagnosis of priority and treatment harder and by varying the moment of introduction of the second fire (e.g. after diagnosis of the first fire or during diagnosis). The data showed that an increasing level of complexity had no influence on when the second fire was detected.

Pilots and Cognitive Lockup

The most famous example of cognitive lockup comes from the aviation domain. In 1972 a plane from Eastern Air Lines, flight 401, crashes. During the landing the pilot is warned about a problem with the landing gear. He cancels the landing and sets the plane in autopilot so that he can solve the problem. Unfortunately, due to his occupancy with the landing gear, the pilot missed the warning signals (alarms and air-traffic control) about decreasing altitude, and the plane crashed (NTSB, 1973).

Modeling of Cognitive Lockup

Cognitive Architecture

Cognitive architectures were established in the early eighties as research tools to unify psychological models of particular cognitive processes (Newell, 1994). These early models only dealt with laboratory tasks in non-dynamic environments (Anderson, 1993; Newell, Rosenbloom, & Laird, 1989). Furthermore, they neglected processes such as multitasking, perception and motor control that are essential for predicting human interaction with complex systems in highly dynamic environments like the air traffic environment addressed in HUMAN with the AFMS target system. Models such as ACT-R and SOAR have been extended in this direction (Anderson et al., 2004; Wray & Jones, 2005) but still have their main focus on processes suitable for static, non-interruptive environments. Below we provide a short overview of the requirements we have for the cognitive model and how these requirements are met by ACT-R 6.1.4, SOAR 9.3.0 and EPIC. Note that we evaluate the requirements only for these versions. ACT-R and SOAR are under constant development and requirements that are not met at the moment might be met in future versions.

The first requirement is that the cognitive model should support multitasking. The three best known cognitive architectures all support a form of multitasking; ACT-R with threading (e.g. Salvucci & Taatgen, 2008), to SOAR (Newell, Rosenbloom, & Laird, 1989) and EPIC (Meyer & Kieras, 1997) it is inherent to the architecture. Secondly, because we want to test interfaces there is a need for perception and motor action abilities. This is inherent to EPIC (Meyer & Kieras, 1997), ACT-R is able to do this since ACT-R/PM (Byrne, 2001), and SOAR cannot do this without coupling with EPIC, although since SOAR 9 there is a vision module (Laird, 2008). All three need interface coupling with a model of the interface (e.g. developed with SegMan (Amant et al., 2005)). Thirdly, the model should be able to learn, SOAR and ACT-R are able to learn, but EPIC is not. Fourthly, we want an explicit Skills-Rules-Knowledge separation (Rasmussen, 1983) to make it easier for users to choose a level on which they want to work and to make it more clear for end users where errors came from. When it is from rules (procedures), adapting procedures can be a solution, when it comes from the knowledge level the solution can be more difficult, because the problems that arise from this level are inherent to people. Finally, it is very important that non-expert users can use the cognitive model in the design and testing process of interfaces. With none of the three discussed cognitive architectures this is possible, because they all require a high level of knowledge of the model, in addition to programming skills, before being able to adapt them to a certain domain or interface.

In the following, we describe shortly the architecture used in the HUMAN project. We choose to describe the architecture to show that our theory of cognitive lockup is embedded in a broader concept. However, this description will only be short and will not go into (implementation) details, as for the theory of cognitive lockup, these details are not necessary.

The cognitive architecture CASCaS (Cognitive Architecture for Safety Critical Task Simulation) is used to model the cognitive process described in the previous section. For a more detailed description of the CASCaS architecture see Lüdtke et al. (2009). CASCaS has multitasking abilities, has a perception and motor module, is able to learn (e.g. production compilation), has a skills, a rules (associative layer) and a knowledge (cognitive layer) based level. Finally, only when you really want to change something of the architecture programming skills are necessary. Otherwise there are editors for the procedures (domain knowledge) and for the interface description. The procedure editor (Frische et al., 2009) can be used by any domain expert, which has been shown by an informal review that was performed by one of the end user partners in the HUMAN project. And UsiXML (Limbourg et al., 2005) which describes the interface in a way that it can be used by the model can automatically transfer HTML pages into the right format, has a graphical editor so that interface designers can use tools that are similar to what they know and XML programming is also possible. UsiXML is developed by human factor experts at the Belgian Laboratory of Computer-Human Interaction (BCHI).

The core of CASCaS is formed by the layered knowledge processing component that contains the associative and the cognitive layer.

A task that is encountered for the first time is processed on the cognitive level with maximal cognitive effort. This processing is goal driven; alternative plans to reach a goal are evaluated usually through mental simulation, and finally one plan is selected to be executed. With some experience, the associative level is used, where solutions are stored that proved to be successful; the pilot has for example learned how to handle the cockpit systems in specific flight scenarios. According to Rasmussen (1983), processing is controlled by a set of rules that have to be retrieved and then executed in the appropriate context. On the autonomous level routine behavior emerges that is applied without conscious thought, e.g. manually maneuvering an aircraft. When solving a task, people tend to apply a solution on the lower levels first, and only revert to solutions on higher levels when lower-level ones are not available (Rasmussen, 1983) or when the situation requires very careful handling due to unusual and safety relevant conditions.

The associative layer selects and executes rules from long-term memory. It is modeled as a production system. Characteristic for such systems is a serial cognitive cycle for processing rules: A goal is selected from the set of active goals (Phase 1), all rules containing the selected goal in their goal-part are collected and a short-term memory retrieval of all state variables in the Boolean conditions of the collected rules is performed (Phase 2). If a variable is absent in memory, a dedicated percept action is fired and sent to the percept component to perceive the value from the environment and to write it into the short-term memory. After all variables have been retrieved, one of the collected rules is selected by evaluating the conditions (Phase 3). Finally the selected rule is fired (Phase 4), which means that the motor and percept actions are sent to the motor and percept component respectively and the sub-goals are added to the set of active goals. This cycle is started when a Boolean condition of a reactive rule is true. In Phase 2 reactive rules may be added to the set of collected rules if new values for the variables contained in the State-Part have been added to the memory component (by the percept component). In Phase 3, reactive rules are always preferred to non-reactive rules. The cognitive cycle is iterated until no more rules are applicable.

The cognitive layer reasons about the current situation and makes decisions based on this reasoning. Consequently, we differentiate between a decision-making module, a module for task execution and a module for interpreting perceived knowledge (sign-symbol translator). In the following, we will describe the decision-making module in more detail, as it is relevant to modeling cognitive lockup. For more information on the cognitive layer see Lüdtke et al. (2009).

The decision-making module determines which goal is executed. Goals have priorities, which depend on several factors: goals have a static priority value that is set by a domain expert. In addition, priorities of goals increase over time if not executed. Implicitly, temporal deadlines are modeled in this way. If, while executing a goal, another goal has a distinctively higher priority than the current one, the execution of the current goal is stopped and the new goal is attended to. This decision depends on the priorities of the goals and is extended by the parameter *Task Switching Costs* (TSC), which determines the difference the priorities need to have to halt the execution of a goal to select a different goal to be executed. TSCs are described extensively in literature (e.g. Jersild, (1927); Rogers & Monsell (1995)). The higher the TSC is, the higher the priority of another goal needs to be to switch to that goal. To determine whether a goal should be interrupted and a different goal should be executed, the TSC is added to the current task priority. Only if a priority of another active goal is above this threshold, this other goal is chosen to be executed. For a visualization of the goals see Figure 2.

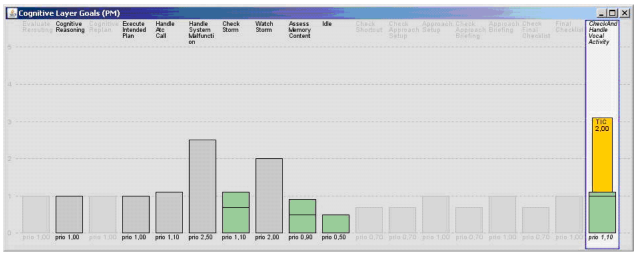

Figure 2: Visualization of the goals on the cognitive layer. Dark gray and green goals are active. The framed goal is currently executed. The yellow staff represents the additional task switch costs.

Cognitive Lockup Model

In this section we describe how cognitive lockup is modeled in the cognitive architecture described above. We model cognitive lockup on the cognitive layer. The main reason for this is that, as described above, on the cognitive layer we have an explicit goal decision mechanism in which cognitive lockup can easily be integrated. However, this can be extended to the associative layer, as the principles explained below are generally applicable to the goals of the associative layer as well.

Time Pressure As described in Neerincx (2003), the time pressure for a person plays an important role for cognitive lockup. If a person has a value for the time pressure of more than 0.75 (Neerincx, 2007), the task switch cost increases. In general, this factor depends both on the time pressure of the associative and cognitive layer. However, to simplify matters, we will model this temporarily only related to the cognitive layer, but will extend the concept later to the associative layer. As written above, the formula that we use is the following:

$$\textit{Time pressure} = \frac{\textit{time required for tasks}}{\textit{time available for tasks}}$$

For example, if we have a task that can be done in 25 seconds and we have 100 seconds before it needs to be finished, the predicted time pressure is 0.25.

The time required for a task is the time needed for cognitively processing the task. This knowledge comes both from the analysis of normative behavior, i.e. discussions with experts that give an indication of the time a task takes, in addition to cognitive theories on which the cognitive architecture is based (e.g. (Anderson, 1993; Kieras & Meyer, 1997)).

Modeling the time that is available for a task is quite complex. For some tasks this knowledge is given in the normative behavior. For example, a pilot needs to have set the flaps before reaching the final approach phase. The time that is available for a task can thus be calculated by the knowledge of the current task, and a prediction of when the approach phase begins, which can be gained from the environment. For other tasks, it is not that easy to know the time that is available to execute it. For example, for a monitoring task, there is no standard deadline at which monitoring has to be finished. However, the time pressure will slowly increase, without having a clear deadline of the task, as there is no unlimited time to execute any task.

Thus, for each task, it has to be evaluated whether the time pressure can be based on a calculation of elements of task knowledge and the environmental input, or whether it has to be given a general estimate.

The time pressure is inherent to each goal as it only takes aspects of the individual goal into account, but is dynamic as the time until it needs to be finished is constantly diminishing. We decided that this calculation is done each 50 ms, which is the cycle time of our architecture.

Level of Information Processing As described above, the level of information processing does not play a relevant role for cognitive lockup. This factor is not taken into account in the model of task switching costs.

Task Set Switches As described above, task set switches are defined as possible goal switches at a given moment. The number of task sets is modeled as the number of goals that are active at the moment. Temporarily, we only look at goals in the cognitive layer.

The value of the task set switches is thus the number of active goals in the environment. We assume that the model always has activated all possible tasks that play a role at the moment in the environment and are needed to handle the current situation.

The Model

Above, we have described different aspects that increase the probability of cognitive lockup. In our model, this is simulated by increasing the task switch costs (TSCs) of the goal that at that moment is processed. The TSC determines the difference that the priorities need to have to halt the execution of a goal to select a different goal to be executed. The TSC depends on the number of goals that at that moment is also active and could be selected to be processed, and on the time to spare to execute the current goal. The TSC is higher when there is high time pressure. Furthermore, the higher the number of active goals is (i.e. the possible task set switches) the higher are the costs to switch to another goal. The following formula determines the TSC:

$$TSC = StartTSC * (Time\ pressure + Task\ set\ switches),$$
$$\text{with } Time\ pressure = 0 \text{ if } Time\ pressure < 0.75.$$

This means that the task switch costs depend on a start value, which is a constant, and the sum of the two factors of the time pressure and the task set switches.

As at each moment if there are active goals, at least one goal is selected and executed, the task set switches parameter is always at least 1. If there is only one goal, and the task pressure is not high, the TSC is equal to the constant start value. The moment there are several active goals or the time pressure for the currently selected goal is above the threshold of 0.75, the TSC is increased.

Conclusion

This paper presented the development of a cognitive model of cognitive lockup: the tendency of humans to deal with disturbances sequentially, possibly overseeing crucial data from unattended resources so that serious task failures can appear—e.g., in a cockpit or control centre. The model is based on real life examples of cognitive lockup and the psychological theories that are derived from these examples, and laboratory experiments. It distinguishes time pressure and number of tasks-to-do as two factors that increase task switch costs and the corresponding risk of cognitive lockup. A heightened task switch cost leads to less task switching, even when another task has a higher priority, as the difference between the priorities needs to be higher.

The proposed model should support the design and evaluation of user interfaces that prevent such failures, being used outside the academic community. The CASCaS architecture proved to best fit with the requirements to incorporate these factors and to support the UI engineering process.

At the moment, we calculate the time pressure as a value inherent to the individual goal. The interdependencies between the timing of several goals will be taken into account in the next version of the cognitive model (i.e., several tasks might in themselves not have a high time pressure, but might together be time-critical, as all of them might need to be finished before all of them can be executed).

The values for the parameters we have chosen for our cognitive model are mainly based on literature, and are currently being evaluated in both laboratory experiments and realistic simulator experiments. In this way, we refine and validate the model, improving its plausibility and predictions about the behavior of pilots. Application of the model will provide user interfaces and procedures that reduce the risks for lockup errors. Due to the cognitive plausibility, we predict that the model can also be used in other domains without substantial changes.

Acknowledgments

The work described in this paper is funded by the European Commission in the 7th Framework Programme, Transportation under the number FP7 – 211988.

References

Amant, R.S. and Riedl, M.O. and Ritter, F.E. and Reifers, A. (2005). Image processing in cognitive models with SegMan. *Proceedings of HCI International 2005.*

Anderson, J. (1982). Acquisition of cognitive skill. *Psychological review*, 89(4), 369–406.

Anderson, J. (1993). *Rules of mind*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Anderson, J., Bothell, D., Byrne, M., Douglass, S., Lebiere, C., & Qin, Y. (2004). An integrated theory of the mind. *Psychological Review*, 111(4), 1036–1060.

Beevis, D., Bost, R., Döring, B., Nordø, E., Oberman, F., Papin, J., et al. (1994). *Analysis techniques for man-machine system design*. AC/243(Panel 8) TR/7 Vol, 2.

Byrne, M.D. (2001). ACT-R/PM and menu selection: Applying a cognitive architecture to HCI. *International Journal of Human Computer Studies*, 55(1), 41-84.

De Jong, R. (1995). The role of preparation in overlapping task performance. *The Quarterly journal of experimental psychology*. A, *Human experimental psychology*, 48(1), 2.

Dekker, S. (2003). Failure to adapt or adaptions that fail. *Applied Ergonomics*, 34(3), 233–238.

Frische, F. and Mistrzyk, T. and Lüdtke, A. (2009). Detection of Pilot Errors in Data by Combining Task Modeling and Model Checking. *Human-Computer Interaction--INTERACT 2009*, 528-531.

Jersild, A. (1927). Mental set and shift. *Archives of Psychology*. Vol, 14(89), 81.

Kerstholt, J., Passenier, P., Houttuin, K., & Schuffel, H. (1996). The effect of a priori probability and complexity on decision making in a supervisory control task. *Human Factors*, 38(1), 65–78.

Kieras, D. E., & Meyer, D. E. (1997). An overview of the epic architecture for cognition and performance with application to human-computer interaction. *Hum.-Comput. Interact.*, 12(4), 391–438.

Laird, J.E. (2008). Extending the Soar cognitive architecture, Artificial General Intelligence 2008: Proceedings of the First AGI Conference.

Limbourg, Q. and Vanderdonckt, J. and Michotte, B. and Bouillon, L. and López-Jaquero, V., (2005). *Engineering Human Computer Interaction and Interactive Systems*, 200-220.

Lüdtke, A., Osterloh, J.-P., Mioch, T., Rister, F., & Looije, R. (2009, September 23–25). Cognitive modelling of pilot errors and error recovery in flight management tasks. In P. Palanque, J. Vanderdonckt, & M. Winckler (Eds.), *Human error, safety and systems development*, 7th ifip wg 13.5 working conference, hessd 2009 (Vol. 5962). Brussels, Belgium: Springer.

Meij, G. (2004). *Sticking to plans: capacity limitation or decision-making bias?* Doctoral dissertation, Department of Psychology, University of Amsterdam, Amsterdam.

Meyer, D.E. and Kieras, D.E. (1997). A computational theory of executive cognitive processes and multiple-task performance: I. Basic mechanisms. *Psychological Review*, 104 (1), 3-65.

Moray, N., & Rotenberg, I. (1989). Fault management in process control: eye movements and action. *Ergonomics*, 32(11), 1319–1342.

NTSB (1973). Eastern Airlines l-1011, Miami, Florida, December, 29, 1972 (Tech. Rep. No. NTSB-AAR-73-14). Washington, DC: National Transportation Safety Board (NTSB).

Neerincx, M. (2003). Cognitive modelling of pilot errors and error recovery in flight management tasks. In E. Hollnagel (Ed.), *Handbook of cognitive task design* (pp. 283–306). CRC.

Neerincx, M. (2007). Modelling cognitive and affective load for the design of human-machine collaboration. *Lecture Notes in Computer Science*, 4562, 568.

Newell, A. (1994). *Unified theories of cognition*. Harvard Univ Pr.

Newell, A., Rosenbloom, P., & Laird, J. (1989). Symbolic architectures for cognition. In M. Posner (Eds.), *Foundations of cognitive science* (pp. 93–131). Cambridge, MA: MIT Press.

Olsen, D., & Goodrich, M. (2003). Metrics for evaluating human-robot interactions. In *Proceedings of permis* (Vol. 2003).

Rasmussen, J. (1983). Skills, rules, knowledge: Signals, signs and symbols and other distinctions in human performance models. *IEEE Transactions: Systems, Man and Cybernetics*, SMC-13(3), 257–266.

Rasmussen, J. (1986). *Information processing and human machine interaction: An approach to cognitive engineering*. Elsevier Science Inc. New York, NY, USA.

Rogers, R., & Monsell, S. (1995). Costs of a predictable switch between simple cognitive tasks. *Journal of Experimental Psychology-General*, 124(2), 207–230.

Rubinstein, J., Meyer, D., & Evans, J. (2001). Executive control of cognitive processes in task switching. *Journal of Experimental Psychology Human Perception and Performance*, 27(4), 763–797.

Salvucci, D.D. and Taatgen, N.A. (2008). Threaded cognition: An integrated theory of concurrent multitasking. *Psychological Review,* 115(1), 101-130.

Schumacher, E., Lauber, E., Glas, J., Zurbriggen, E., Gmeindl, L., Kieras, D., et al. (1999). Concurrent response-selection processes in dual-task performance: Evidence for adaptive executive control of task scheduling. *Journal of Experimental Psychology: Human Perception and Performance*, 25, 791–814.

Wray, R., & Jones, R. (2005). An introduction to soar as an agent architecture. In R. Sun (Ed.), *Cognition and multiagent interaction: From cognitive modeling to social simulation* (pp. 53–78). Cambridge University Press.