Cognitive Architecture and Modeling Idiom: An Examination of Three Models of the Wickens's Task

Yannick Lallement (yannick@cs.cmu.edu)
Bonnie E. John (bej@cs.cmu.edu)
Human Computer Interaction Institute
Carnegie Mellon University
5000 Forbes avenue, Pittsburgh, PA 15212 USA

Abstract

Cooper and Shallice (1995) raise many issues regarding the unified theories of cognition research program in general, and Soar in particular. In this paper, we examine one specific criticism of Newell's (1990) treatment of immediate behavior and use it to explain the notion of the *modeling idiom* within a cognitive architecture. We compare a dual-task model using Newell's architecture and idiom to two other models that use different architectures and idioms (EPIC and an experimental version of Soar). We also look at the models' dependency on their respective cognitive architectures, and the theory/implementation gap also identified by Cooper and Shallice (1995).

Introduction

Unified theories of cognition seek to provide a unique, consistent framework for modeling all types of cognitive, perceptual, and motor activities. Several unified theories, or *cognitive architectures*, have been proposed: among them are Soar (Newell, 1990) , ACT-R (Anderson, 1993) and EPIC (Meyer & Kieras, 1995). Cooper and Shallice (1995) raise many issues regarding the unified theories of cognition research program in general, and Soar in particular (although we believe these objections apply to other architectures as well). In this paper, we examine some of Cooper and Shallice's issues by modeling dual-task behavior.

Newell (1990) described an idiom, or style of modeling that is *possible* within an architecture but not *required* by the architecture, called PEACTIDM for immediate behavior tasks. PEACTIDM is an acronym for the operators the idiom allows at the immediate behavior level (without problem-solving): ***P***erceive, ***E***ncode, ***A***ttend, ***C***omprehend, ***T***ask, ***I***ntend, ***D***ecode, and ***M***otor, (pronounced pee-ack-ti-dim). According to Cooper and Shallice (1995), Soar/PEACTIDM "would seem to provide only a rigidly single channel mode of operation..." and, therefore, in a model of a dual task "...the primary task will inevitably be slowed by up to 120 ms" (p. 134-135), which is not confirmed by data they cite. To investigate the criticism, we chose a specific dual task that allows us to compare a Soar/PEACTIDM model to two other existing models of the task: one written in EPIC and one in an experimental version of Soar with a different idiom.

In addition to examining Soar/PEACTIDM's appropriateness for immediate behavior, we also discuss the dependence of each model's behavior on its underlying cognitive architecture, and the theory/implementation gap identified by Cooper and Shallice.

The Wickens's Task & Modeling Results

The Wickens's task (Martin-Emerson & Wickens, 1992) illuminates phenomena associated with heads-up displays in aviation. It consists of a continuous tracking task interrupted by a choice-reaction task. The Wickens's task display is shown in Figure 1. The upper part of the display shows a target and a cursor in the tracking window. The cursor moves randomly and the participant must keep the cursor inside the target by moving a joystick with his right hand. While the participant performs the tracking task, a left- or right-pointing arrow appears at random intervals in the information display, below the tracking window. The stimulus is displayed for one second. The participant must reply to this stimulus by pressing one of two keys with his left hand, while still attempting to keep the cursor in the target. Experiments were run with different separations of the target and information display to study how this affects two performance measures. *Tracking error* is the RMS error measured only during the two seconds after the stimulus is displayed. *Reaction time* is the time difference between the onset of the stimulus and the response of the user.

Two previous models of this task, one written in EPIC, with the PPS cognitive processor (Kieras & Meyer, 1995) and one written in an experimental version of Soar (called "Soar-Operand") using EPIC's peripherals (Chong & Laird, 1997), give results very close to human performance. We wrote a third model, based on the existing models, using the current Soar7 release and Newell's PEACTIDM idiom, which also matches well (Figure 2). For each model, each

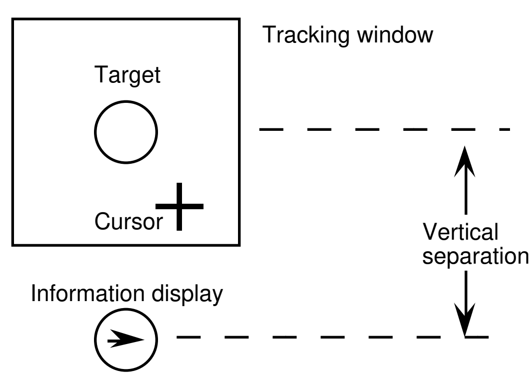
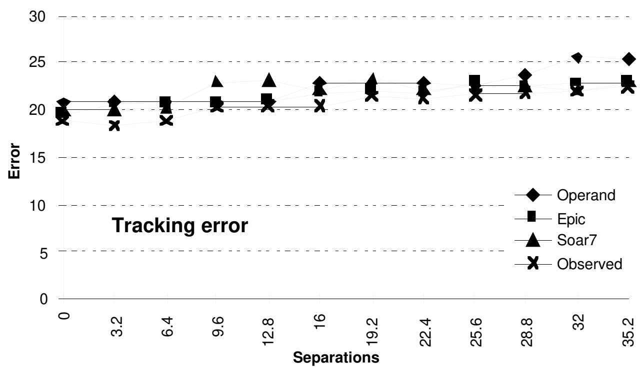


Figure 1: The Wickens's task environment

point on Figure 2 represents an average over 300 trials. The observed performance is an average over 24 participants. The absolute average error for all three models is always less than 10% for both the choice and the tracking tasks.

Clearly, the Soar/PEACTIDM model is no worse than the other two models, which do not share the "rigidly single channel mode of operation" that worried Cooper and Shallice. In examining the commonalities and differences between these models, we will illustrate two other issues brought up by Cooper and Shallice, how little the behavior of these models depends on the *cognitive* aspect of the architecture alone, and the theory/implementation gap.



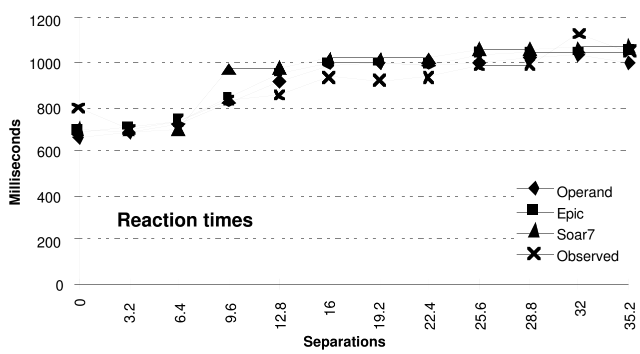


Figure 2: tracking error and reaction times of the three models and observed human data

The Three Models

Commonalities of the Models

EPIC Perceptual and Motor Processors EPIC is a cognitive architecture originally intended to model multiple tasks performance (Meyer & Kieras, 1997). It is composed of a specific cognitive processor, PPS (discussed below) and of a set of perceptual and motor processors. EPIC's perceptual and motor processors are used for all three models. These processors are native to EPIC and were adapted to communicate with Soar's cognitive processor by Chong and Laird (1997). This union answered a common criticism of Soar (e.g., Cooper & Shallice, 1995; Vincente & Kirlik, 1992) that it does not take motor and perceptual constraints into account.

The perceptual and motor processors communicate with the cognitive processor via the working memory (WM). They work independently, in parallel with each other and with the cognitive processor in all the models. The Wickens's task models use the visual perception, the tactile perception, the oculomotor and the manual-motor processors. These processors do not perform any actual information processing; they provide precise timings for the functions they simulate. For example, when an object appears on EPIC's "retina", its location will be reported by the visual perceptual processor to WM with a 50 ms latency, and if the object is in EPIC's "fovea", its shape will be appear in WM after an additional 50 ms latency (Kieras & Meyer, 1996). These delays mimic those of the human processes for localization and shape-recognition, but the features are provided as input to the visual processor by the simulation of the environment.

Some processors can perform actions by themselves; for example the oculomotor processor can produce small centering saccades to keep a slowly moving object in the fovea. The motor processors "jam" if they receive more than one command at a time, i. e., they will ignore all of them. The manual-motor processor does not require cognition to calculate a specific direction to push the joystick (pushing the joystick in any direction is assumed to take the same time). However, it requires the name of the button to press (LEFT or RIGHT) because the finger may or may not have to move to the new button, changing the time to execute the operation.

Finally, the motor processors have a separate preparation and execution phase. That is, under certain conditions, the preparation of a motor movement can be overlapped with the execution of the previous movement. In addition, the preparation phase can be skipped altogether if a movement is identical to the immediately previous movement (Kieras & Meyer, 1996).

Eye-Movement Hypothesis All three models are based on the assumption (formulated by Kieras & Meyer, 1995) that the eye must be kept on the cursor to ensure successful tracking, and that the eye must be moved to the choice stimulus in order to discriminate it.

Task Simulation The simulation of the task is done in an EPIC "device processor" for all three models. This processor informs the perceptual processors of changes in the environment (e.g., the onset of the stimulus), and responds to motor commands from the models.

Human Data Finally, all three models are matched against the same human performance data. The data is aggregated over 24 participants who were instructed to perform as well as they could. The collection procedure is described in (Martin-Emerson & Wickens, 1992).

Differences Between the Models

Execution Traces for the Three Models Figure 3 shows the execution traces of the three in PERT charts. The models have been tracking the cursor prior to this figure and the first stimulus comes up at what we've labeled 0 ms.

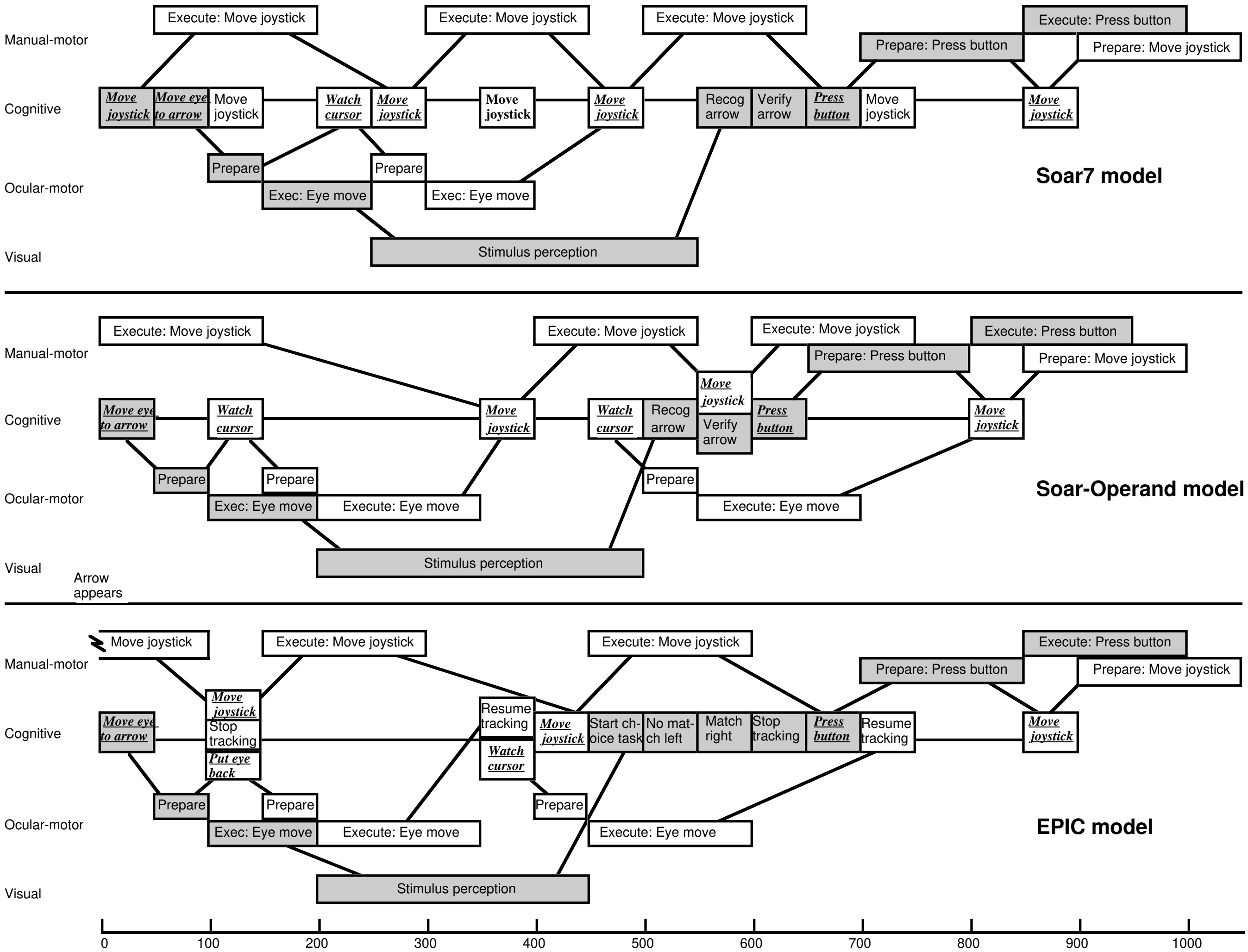


Figure 3: the execution traces of the three models. The critical path for the answer task is grayed.

Complete operation of the manual-motor and oculomotor processors are shown, as well as visual perception delays. Only critical operations of the cognitive processors are shown (described below). Cognitive operations that send commands to the motor processors are in italics and underlined. The gray boxes denote the *critical path* (the longest path through the task) for the choice reaction-time task.

Not surprisingly, given the good fit to the human data, the three models take similar motor actions at similar times (though each model has a slightly different initialization phase and are therefore not in perfect synch on the tracking task). More precisely, during the 1s interval shown in Figure 3, each model moves the joystick three times, and the motor actions that respond to the choice task are executed with at most one cycle difference. However, their internal cognitive operations are different, reflecting the different cognitive processors and idioms they use.

Cognitive Processors and Idioms Each model uses a different cognitive processor, Soar7, Soar-Operand, or EPIC(PPS) and a different modeling idiom. All three cognitive processors are production systems and each decision cycle is estimated to approximate 50 ms of human behavior. The cognitive processors and idioms differ on other aspects listed below.

Soar7's cognitive processor and the PEACTIDM idiom. In Soar (both Soar7 and Soar-Operand), an operator is implemented by a set of fine-grain productions that propose, apply and terminate it. At the end of each decision cycle, an operator is selected, which will typically be applied (i.e., make changes to WM) at the beginning of the next decision cycle. Operators that request motor actions, however, may not be applicable in the very next decision cycle because the peripheral processor may be busy completing the last request. Soar's architecture allows the application of an operator to be separated in time from its selection, as long as nothing intervenes to make its conditions invalid. The distinction is shown in the Soar7 model in Figure 3 by plain font (motor-operator-selection without immediate application) and italics (motor-operator-application either immediately after its selection or from a previously-selected and still valid operator).

If appropriate knowledge is not available to select or to apply an operator, an impasse occurs, and a new state is created where more knowledge can be brought into play. When the impasse is solved, Soar's learning mechanism creates a new production that associates the conditions under which the impasse occurred with the actions that resolved it. If this situation arises again, the newly learned production can often prevent another impasse. In Figure 2 (to avoid clutter) impasses are implicit in the gaps between cognitive operators. For instance, the gap between a plain-font MOVE-JOYSTICK operator and its italicized application is an impasse, where the "missing knowledge" is that the manual processor is free and the "solution" is to wait for the manual processor to finish what it was doing. This type of impasse can never be eliminated through learning; it is the Soar architecture's response to a constraint dictated by the peripheral processors.

The Soar7 model has no explicit control process. The actions to be taken are determined by the state of current perceptions and motor processor states in WM. There is no explicit record of which procedural steps have been accomplished.

The Soar7 model uses the PEACTIDM idiom proposed by Newell (1990) for immediate behavior tasks. Newell successfully applied this idiom on several such tasks, but no dual tasks. The PEACTIDM idiom assumes that reaction to events in the real world begins with a ***P***erceive operation that is handled by EPIC's perceptual processors. ***E***ncode productions translate the signals delivered by EPIC's perceptual processors into meaningful symbols in WM (not shown in Figure 3). Once in central cognition, the PEACTIDM idiom allows only four types of elementary cognitive operators to be selected and applied without impasse: ***A***ttend operators direct perception to a new stimulus (e.g. WATCH-CURSOR); ***C***omprehend operators interpret the perceptual inputs (e.g. RECOGNIZE-ARROW); ***T***ask operators select the next task to be done (none appear in Figure 3; these happen at the start of the trial, before this timeline begins); ***I***ntend operators initiate a motor response (e.g. MOVE-JOYSTICK). After central cognition has intended a motor command, ***D***ecode productions translate this command into symbols recognized by the EPIC motor processor (not shown in Figure 3), which then perform the ***M***otor actions.

A Soar model using the PEACTIDM idiom sends output commands sequentially from central cognition to the motor processors, no more than one command per decision cycle. Therefore, there is no risk of jamming the EPIC peripherals.

Our Soar7 model learns when to propose motor operators. Before learning, the model knows the available motor operators, but not when they are applicable in the real world. Therefore, it generates an impasse at each decision cycle and reasons about operators in an internal "imagined" world to determine which operator to propose. After learning, it knows when each action is applicable and proposes appropriate operators without impasse. Figure 3 shows performance after learning.

Soar-Operand's cognitive processor and the hierarchical-operator idiom. Soar-Operand is an experimental version of Soar that was developed in response to a perceived difficulty with the interaction between Soar's learning mechanism, action in the real world, and a modeling idiom that uses a hierarchical operator structure (John, 1996).[1] In certain cases, a model with a hierarchical operator structure would learn to perform strings of actions that were no longer appropriate by the time they were learned. To avoid this problem, Soar-Operand is restricted to making only one round of persistent changes to WM per decision cycle. Although multiple persistent changes to WM can be made in a single decision cycle, no change can depend on a previous change being made (i.e., no strings of changes are allowed). Since initiating a motor action requires a persistent change to WM, this restriction affects the behavior of dual tasks like the Wickens task.

As with the Soar7, the Soar-Operand model has no executive control. The hierarchical-operator modeling idiom used with Soar-Operand proposes a DUAL-TASK operator that is always selected in this task. Before learning, the model does not know how to apply the DUAL-TASK, so an impasse arises. Under different perceived conditions in the task environment, the model learns to apply the DUAL-TASK operator by taking different motor actions (watching the cursor, moving the joystick, etc.). After learning, the same operator is always selected, but the automatic application of that operator differs depending on the current environmental conditions.

With this approach, the Soar-Operand model can make simultaneous requests to the same motor processor and jam EPIC's peripherals. This conflict also causes an impasse to arise and, using task knowledge (i.e., the choice task has priority over the tracking task), the Soar-Operand model learns how to avoid jamming. Figure 3 shows the *applications* of the DUAL-TASK operator along the Cognition line. The selection of the DUAL-TASK operator itself is not shown because it is ubiquitous and its differing applications display the model's actual behavior.

EPIC's PPS cognitive processor and the executive control idiom. PPS (parsimonious production system) is the cognitive processor used by the EPIC model. PPS is a fully parallel production system: as soon as the conditions of a production rule are satisfied by the contents of the WM, its actions will be executed, without any conflict resolution mechanism. Any number of productions can be executed simultaneously. An operation in PPS (such as issuing a command to a motor processor) is implemented by one single production, so PPS's *productions* are similar to Soar's *operators*. PPS has no learning mechanism.

The EPIC modeling idiom emphasizes the executive process that coordinates between the two tasks. Executive-process productions explicitly keep track of which procedural step is currently being worked on in order to decide on the next procedural step or to switch to a task with higher priority. In addition, executive control knowledge is hand-coded so two motor commands will never be sent to the same motor processor. Thus, the EPIC model's peripherals

[1] In contrast, PEACTIDM uses a flat operator structure consisting only of the A, C, T & I operators described above. The problem does not arise in the models using the PEACTIDM idiom.

never jam. Kieras and Meyer (1995) show how a strongly interleaved executive control was necessary to obtain good match to the data.

Parallelism issues The main difference in the three architecture/idiom combinations is how they handle parallelism. Both PPS and Soar-Operand allow their models to initiate several task-related actions in a single cycle: for example the EPIC model sends two motor commands at time 100 ms; the Soar-Operand model sends a command and performs an internal cognitive operation at time 550 ms (Figure 3).

Although Soar7/PEACTIDM has completely sequential operator selection it still provides a good matching to the data. This can be explained by the strong interruptibility of this model accomplished through Soar's separation between selection and application of an operator. For instance, at 100 ms, the Soar7/PEACTIDM model initiates a MOVE-JOYSTICK command but it cannot be applied right away. Two decision cycles later the model realizes that its eye is no longer on the cursor, so it should not move the joystick to try to track the cursor. It interrupts the MOVE-JOYSTICK operator in favor of a WATCH-CURSOR operator. Thus, a highly-interruptible Soar7/PEACTIDM model can function as effectively in this task as the other more parallel models. Only at two cycles in EPIC, and one cycle in Soar-Operand is the parallel-action feature of the respective architectures used. In the end, it appears that parallelism in cognition is not required to perform even as rapid and tightly measured a task as the Wickens task.

Discussion

The PEACTIDM Idiom

Contrary to Cooper and Shallice's (1995) concern that the PEACTIDM idiom's single channel mode of operation would be inappropriate for dual-task situations, our Soar7/PEACTIDM model provides as good fit to the data as the more parallel Soar-Operand and EPIC models. We believe this is evidence that the PEACTIDM idiom, sequential though it may be, should not be dismissed out of hand for dual-tasks.

As noted in previous work with complex real-time tasks (e.g., Nelson, Lehmann & John, 1994), the behavior of the PEACTIDM idiom in this model can be explained by the small granularity of the operators, by their tight interleaving, and by the feature of the Soar architecture that operator selection and operator application can be separated. This enables the selection of another operator before the application of the current one is completed, as well as the ability to be interrupted in reaction to changes in the environment. Thus, the PEACTIDM idiom provides more parallelism that Cooper and Shallice realized.

Exploiting the selection/application separation capability of the Soar architecture is essential to obtaining a good fit to the human data with the PEACTIDM idiom. A preliminary version of our model, which differed from the final version only in that it did not exploit this separation, had a tracking-error twice as high as reported here. Although we believe our model demonstrates that the Soar architecture with the PEACTIDM idiom is more flexible than Cooper and Shallice thought, this also demonstrates another of Cooper and Shallice's points. The complexity of these architectures (and Soar is not alone) requires a substantial effort, programming as well as reading the theory, to make valid pronouncements of what they can and cannot do.

Role of the Architecture and the Idiom

In the case of these specific models, it seems that the choice of the architecture and of the modeling idiom, together with matching human data, imposed enough constraint to lead to specific models whose output is similar. Each model has a history of successive versions that were quite far from human performance, and had to be improved upon. Kieras and Meyer (1995) present an earlier model with a simpler executive process that gave unsatisfactory results (correct reaction times, but tracking error more than 100% higher than human data). Chong and Laird (1997) give the whole path from the first preliminary Soar-Operand models (with results similar to the first EPIC model) to the final model discussed here. They describe successive small modifications (sometimes as small as one condition added to a production) that strongly improved the results. In our own model, the dissociation between selection and application of operators, leading to more interruptibility and interleaving, shrunk the tracking error by more than 50%.

This answers an objection to Soar brought up by Hunt and Luce (1992), that two Soar modelers could come up with two different models of a same task. This is certainly true. For example, a PEACTIDM model could have been implemented in Soar-Operand with the same results as the Soar7 model. Alternatively, executive-process knowledge could be hand-coded into either version of Soar to make those models similar to EPIC's. However, when the architecture/idiom couple is considered, this may not remain true, the field of the possible models seems to be much more restrained. The role of the modeling idiom may turn out to be as important at that of the architecture itself.

Role of the Cognitive Processor

Copper and Shallice noted that performance of Newell's immediate behavior models did not depend heavily on the Soar cognitive architecture. We see that all three models of Wickens's task also suffer from this complaint. In each of the three models, more than a third of the cognitive cycles are spent waiting or doing nothing. For the choice task, where participants were instructed to respond as quickly as possible, cognition is on the critical path (in gray on Figure 3) only 20 to 30% of the time. On the other hand, the perceptual and motor processors (identical for the three models) are the most important contributors. For example, the ascending slope of the tracking error with target separation results from the fact that, in each model, tracking is disabled while the eye moves and it takes more time for the eye to move between the choice stimulus and the cursor when they are further apart.

Our experience modeling immediate behavior in these three architectures leads us to question whether it is possible to find a task that would shed more light on the underlying cognitive architecture. To differentiate between architectures, we need a task with a higher percentage of cognitive operations on the critical path. However, if the task is even faster, but still requires perception and motor response, the behavior of the peripherals will likely dominate. If the task is slower but more complex so more cognition is needed, then this introduces opportunity for individual differences in task execution. Even what seem like rapid tasks, a video-game like the Kanfer-Ackerman ATC task© (John & Lallement, 1997), or sentence verification (Reder, 1988), or a simple arithmetic task (Siegler, 1996), have been shown to have substantial differences in the strategy people use to perform the task. Thus, the effects of an individual's prior knowledge, problem-solving skill, visual-search skill, etc. will dominate over the effects of the underlying cognitive architecture. This might be addressable by making models of individuals rather than of aggregate data, but then the modeling work multiplies intractably.

It appears that finding a single task that would discriminate among different cognitive architectures is not easy, if not impossible. Newell's (1990) solution to this problem is not to look for a single task, but to bring to "bear large, diverse collections of quasi-independent sources of knowledge -- structural, behavioral, functional -- that finally allow the system to be pinned down." (p. 22). If the Wickens's task can be equally well modeled with a set of idiom/architecture pairs, and a lower-level task can be equally well modeled with a different set, and a higher-level problem-solving or learning task with still a third set, then the intersection of these sets may point toward a single idiom/architecture pair that explains this large amount of behavioral data. Thus, multiple models of the same task in different idiom/architecture pairs, and consistent use of an idiom/architecture pair across multiple tasks, both make contributions to such a research program.

Theory/Implementation Gap

Our work also corroborates Cooper and Shallice's point about a theory/implementation gap. Each of these models are very sensitive to details; small changes have big effects and every detail of the implementation is important for the final results. All the details are not always provided by the relevant documentation (especially short conference papers), but reside only in the code itself. For example, Copper and Shallice's worry about a rigid single channel did not materialize within the details of a running Soar program. Also, there is a production in the EPIC model that allows it to bypass some operations relevant to the choice task if the same stimulus appears two consecutive times, saving one cognitive cycle. Being activated in 50% of the cases, this single production lowers the average reaction time by half the duration of a cognitive cycle, or 25 ms (~3% of the observed response time). Our work extends Cooper and Shallice's comments on the gap between the theory and implementation of *architectures* to the individual cognitive *models* themselves. In order to completely understand what a model does and how it does it, every detail counts, and the access to the program code is necessary.

Acknowledgments

Thanks to Ronald Chong and David Kieras for their continuous support with EPIC, EPIC-Soar, and their respective models of the Wickens's task. This research is supported by the ONR, Cognitive Science Program, Contract Number N00014-89-J-1975N158. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the ONR or the US Government.

References

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

Chong, R. S. & Laird, J. E. (1997) Towards learning dual-task executive process knowledge using EPIC-Soar. *Proceedings of the 19th annual conference of the cognitive science society* (pp. 107-112). Hillsdale, NJ: Lawrence Erlbaum Associates.

Cooper, R. & Shallice, T. (1995) Soar and the case for unified theories of cognition. *Cognition*, 55, 115-149.

Hunt, E & Luce, R. D. (1992) Soar as a world view, not a theory. *Behavioral and brain sciences*, 15:3, p 147.

John, B. E. (1996) Discussion: task matters. In D. M. Steier and T. M. Mitchell (Eds.), *Mind matters, a tribute to Allen Newell* (pp. 313-324). Hillsdale, NJ: LEA.

John, B & Lallement, Y. (1997) Strategy use while learning to perform the Kanfer-Ackerman Air Traffic Controller Task. *Proc. of the 19th annual conference of the cognitive science society* (pp. 337-342). Hillsdale, NJ: LEA.

Kieras, D. E. & Meyer, D. E. (1995) *An overview of the EPIC architecture for cognition and performance with application to human-computer interaction* (Tech. report TR-95). Ann Arbor, MI: University of Michigan.

Kieras, D. E. & Meyer, D. E. (1996) *The Epic architecture: principles of operation*. Electronically published at ftp.eecs.umich.edu:/people/kieras/EPICarch.ps.

Martin-Emerson, R. & Wickens, C. D. (1992) The vertical visual field and implications for the head-up display. *Proceedings of the human factors society* (pp. 1408-1412).

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

Nelson, G., Lehmann, J. F., John, B. E. (1994) Experiences in interruptible language processing, In *Proceedings of the 1994 AAAI Spring Symposium on Active Natural Language Processing*, 1994.

Newell, A. (1990) *Unified theories of cognition*. Cambridge, MA: Harvard University Press.

Reder, L. M. (1988). Strategic control of retrieval strategies. In G. Bower (Ed.) *The psychology of learning and motivation*, vol. 22 (pp. 227-259). NY: Academic Press.

Siegler, R. (1996). *Emerging minds: The process of change in children's thinking.* NY: Oxford University Press.

Vincente, K. J & Kirlik, A. (1992) On putting the cart before the horse: Taking perception seriously in unified theories of cognition. *Behavioral and Brain Sciences* 15:3, 461-462.