In N. Taatgen & J. Aasman (Eds.), *Proceedings of the 3rd International Conference on Cognitive Modelling*. 202-209. Veenendaal, The Netherlands: Universal Press.

Extending Tcl-Tk to provide a functional eye and hand for the Soar cognitive modelling architecture

Peter R. Lonsdale (plonsdale@acm.org)
School of Psychology
University of Nottingham
Nottingham NG7 2RD UK

Frank E. Ritter (ritter@ist.psu.edu)
School of Information Sciences and Technology
Penn State University
State College, PA 16801 USA

Abstract

It would extend the world that cognitive models can see if models could interact based on an interface language. We introduce a system designed to allow cognitive models to interact with any display written in Tcl/Tk, a common interface creation language. This approach can be viewed as extending cognitive architectures to include eyes and hands that exist in the world of Tcl/Tk. This system is most naturally used by Soar because Soar includes Tcl/Tk, but the eye/hand could be used by other architectures. We demonstrate an initial model that uses independently created on-screen phones. This model does the task itself and more accurately predicts a task time than GOMS. Findings from the implementation of Soar/Tcl-PM are used to explore current limitations of GOMS-type task analysis methodologies, possibilities for enhancing the communication between Soar and Tcl/Tk, and data gathering requirements for future cognitive modelling in this area.

1 Introduction

Cognitive models can be used to predict performance and thus usability on systems that have not yet been developed. Tools such as the Model Human Processor, the Keystroke Level Model, and the GOMS (Goals, Operators, Methods, and Selections) family of techniques (Card, Moran, & Newell, 1983; John & Kieras, 1996) have all been put to effective use. When they are applied, they can literally save millions of dollars (Gray, John, & Atwood, 1993).

It would be useful to have process models interact with interfaces directly to obtain these results. Having process models interact with interfaces directly would also be good psychology, for it would lead to a wider and richer cognitive architecture. Moving the performance of a task into the task domain itself rather than relying on problem solving in pre-defined problem spaces engenders the consideration of a realistic context in modelling performance of such tasks. The need for contextually-influenced interaction with the environment is acknowledged by cognitive modelling architectures. Newell (1990) lists perceptual and motor processes as a prime consideration for any unified theory of cognition such as Soar. Similarly, Anderson, Matessa, and Lebiere (1998) acknowledge that previous work with cognitive architectures, typically assuming black-box models of perception and action, are inadequate for the modelling of more complex and interactive tasks.

1.1 Models as Users: Interactive Cognitive Models

Models substituting directly for the users of systems must be able to perform tasks of a dynamic and interactive nature. The use of interfaces can be likened to problem solving, but unlike the

In N. Taatgen & J. Aasman (Eds.), *Proceedings of the 3rd International Conference on Cognitive Modelling*. 202-209. Veenendaal, The Netherlands: Universal Press.

task of searching a problem space for a series of operators that achieve the goal, interactions with software are likely to require a model that responds to changes in the state of the world. Implementing models in this way is still an emerging field. Most cognitive models simulate only the cognitive aspects of single tasks and as such they are not as easily generalisable or reusable.

Interactive models, which use perceptual processes to form internal representations of an external task environment, provide the opportunity to study and model the ways that real users perform these low-level perceptual processes. To study human-computer interaction as the communication of states and intents, we need some way of investigating how user intents are expressed and, perhaps more importantly, how the user then perceives the system state. Theories of higher-level cognition have typically ignored lower-level processes such as visual attention and perception (Anderson, Matessa, & Lebiere, 1998). The ability to view a model's performance through an interface that displays the elements being perceived and the actions of the perceptual and motor mechanisms (such as a virtual eye and hand) renders combined theories of cognition, perception, and action more readily observable and testable.

1.2 Existing Interactive Models

The creation of interactive models with psychological claims show that this approach is now tractable enough to be performed and provide several lessons. Ritter, Baxter, Jones, and Young (in press) review several examples (i.e. Bass, Baxter, & Ritter, 1995; Byrne & Anderson, 1998; Jones, Ritter, & Wood, 2000). The review notes that including interaction allows more complex models to be created that interact with more complex worlds, and that these models provide a better approximation to human behaviour.

The models are easiest to develop when the task is simulated in the computer language that is used to implement the model's cognitive architecture. Reuse of the models and tasks is possible and highly desirable, but requires some thought in implementation. It is possible (but not always done) to design the interaction component so that it is possible to interact with most interfaces written in a interface creation tool or language.

1.3 Requirements for Interactive User Models

There are several ways of providing task simulations for cognitive models. The first is to have the task simulation implemented in the cognitive modelling language itself. This approach is best for (and probably limited to) simple tasks where the focus is on strictly cognitive aspects of performance. The second way is to have the model interact with simple function calls to an existing application. The third way is to have a real interface that can be operated by a model and a real user (e.g. Bass et al., 1995). This last approach is complex, involving the need for extensive processing of the interface itself either through manipulating the data structures (Ritter et al., in press) or through visual pattern recognition algorithms (e.g. Zettlemoyer & St. Amant, 1999).

Unless the research requires the modelling of a highly complex task, which cannot be implemented using an interface simulation tool, the third approach described above is probably the best one for generalised implementation and study of interactive user models. Using real interfaces introduces complex issues of giving models the means to perceive and act on the interface; confining the simulation to the model reduces the opportunity of studying the actual interactions taking place between model and interface.

Ritter *et al* (in press) identify a set of requirements for the development of interactive cognitive models using this approach that includes (a) tool to create interfaces, (b) a run-time mechanism that determines how the model will interact with the task simulation, and (c) a linkage mechanism that provides the means for communication between the task simulation and the model. These three requirements are supported by user interface management systems (UIMS).

In N. Taatgen & J. Aasman (Eds.), *Proceedings of the 3rd International Conference on Cognitive Modelling*. 202-209. Veenendaal, The Netherlands: Universal Press.

Soar and ACT-R currently have access to UIMS. In the case of Soar, it is Tcl/Tk (Ousterhout, 1994). Models written in Soar have the potential to interact with a task domain using interfaces, control mechanisms, and linkage mechanisms all implemented in Tcl/Tk. We introduce and test such a system here. Some of the implications of this work are considered along with possible future lines of research.

2 The Soar/Tcl-PM System

Soar/Tcl-PM is an extension of the PracTCL application (Harris, 1999), which allows cognitive models implemented under Soar to be routinely linked to interfaces generated by Tcl/Tk using a sim-eye and sim-hand created in Tcl/Tk.

The simulated eye and hand are in the form of windows bounding what the sim-eye can see and another window indicating the position of the sim-mouse'. Tcl code allowed the passing of information to Soar's working memory about what objects were currently visible under the eye, and Tcl procedures were provided that allowed Soar to move the eye and mouse around the screen. Mouse-clicking is simulated by having Tcl send a mouse press event to the object directly under the window representing the virtual mouse. Harris provided a relatively simple Soar model that generated default scanning behaviour for the eye allowing it to search for a button labelled with the digit 1 and then to press it. We note here the improvements we have recently made to the Soar/Tcl-PM system, particularly its Tcl/Tk interface, and an initial Soar cognitive model to use it.

2.1 Improving the Soar/Tcl-PM Interface

We have continued to extend the usability, functionality, and abilities of Harris's system, now renamed as Soar/Tcl-PM, to further the development of reusable, generalised, and interactive user models. '/Tcl' indicates that it is an extension of Soar to use Tcl. 'PM' represents Perceptual Motor. This is intentionally similar to ACT-R/PM. We believe it will become more similar with further development. Soar/Tcl-PM is designed to support including all the mechanisms and regularities of previous models developed using this approach (Ritter et al., in press), such as a fovea and parafovea. Not all of the regularities have been included and tested.

The improved graphical interface for Soar/Tcl-PM is shown in Figure 1. This interface provides a manual method of operating the eye and hand and a graphic display of its state for debugging and demonstrations. All of the functions and information that a cognitive model can access can be invoked by using the graphical display.

In the top centre is an example phone interface. The sim-eye as defined by four Tcl/Tk windows is centered on the '0' button. The objects it can see when it looks (fixates) are shown in the window on the far top left. The mouse pointer is centered on the '0' button as well, and has just clicked there as noted by the '0' in the display of the telephone.

Control panels on the left side allow the analyst to move (saccade) the eye around the screen, cause the eye to 'look' at whatever is beneath it, and to move and click the mouse. The model performs these task by passing commands to the sim-eye and sim-hand.

2.2 Extending the Cognitive Model: Dialling a Series of Digits

Lonsdale (1999) provided several extensions to the cognitive model controlling Soar/Tcl-PM's eye and hand so that the model could dial a full phone number. The original model was able to search for '1' and click on its button, but there was no facility for then having it search for a subsequent digit to dial or for any general text. Additionally, the default scanning strategy of the eye is to sweep the entire screen in a left to right, top to bottom movement. This strategy is extremely inefficient.

In N. Taatgen & J. Aasman (Eds.), *Proceedings of the 3rd International Conference on Cognitive Modelling*. 202-209. Veenendaal, The Netherlands: Universal Press.

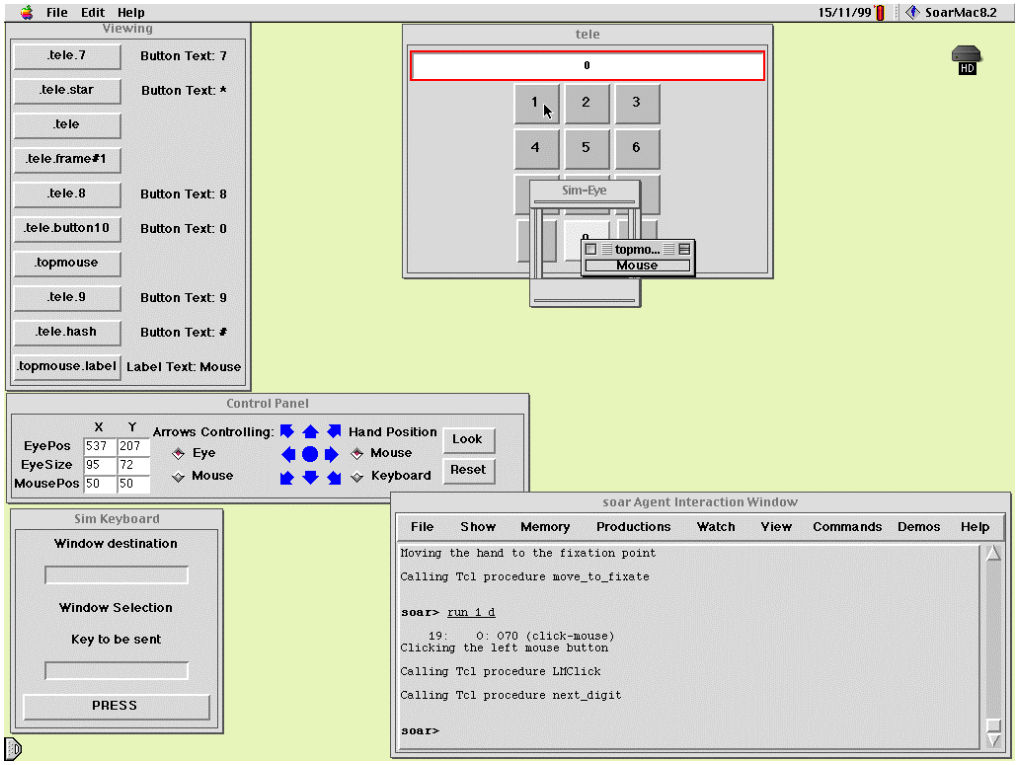

Figure 1. Combined control panel for eye and mouse.

A search strategy has been added that makes use of existing operators to first locate a button with any text string using the default scanning strategy. Once the target button has been found, default scanning productions no longer fire. Instead, another set of operators are applied to produce a sweeping movement across the digit buttons. The eye sweeps the display in the horizontal plane until there are no buttons visible. The eye then steps up or down (depending on the vertical direction in which it last moved) and sweeps in the reverse horizontal direction. If vertical movement results in no buttons being visible, the eye reverses its vertical scanning direction, and resumes scanning horizontally.

Many assumptions were made about the processes involved in human visual perception. These assumptions are not intended to form the proposal or endorsement of any particular theory of human visual perception *per se*. The extensions given to the cognitive model were derived solely from the intention of making the model succeed in performing the desired task of dialling a series of digits. They do, however instantiate an unusual type of visual theory. This theory does not *describe* what processing occurs, but actually *processes visual information*. This processing is a natural first step in building any cognitive model. Comparison to actual human data is the next. The issues surrounding comparison to human data and the relation of the Soar/Tcl-PM's cognitive model to current theories of visual perception are reprised in the conclusion, along with possible methods of gathering actual user data in this area.

Memory is also important. Preliminary runs of the model illustrated that the simple sweeping scan wasted a lot of time searching for digits it had already dialled, such as the second 1 in 1471. It seemed apparent that a real user would not duplicate efforts in this way, and would at least recall the approximate position of the 1 button from having dialled it previously. The model should remember previously spotted items, at least under certain circumstances. For example, the model should remember for a short period of time at least the position of items which

In N. Taatgen & J. Aasman (Eds.), *Proceedings of the 3rd International Conference on Cognitive Modelling*. 202-209. Veenendaal, The Netherlands: Universal Press.

it has fixated. The fixate productions were modified to add elements to the working memory storing the position of any digit button upon which the eye fixated, and additional spotting operators were implemented that could act upon this stored information in preference to scanning the screen for a second time. These results are similar to those of Altmann and John (1999).

As a matter of expedience, Tcl code was used to model retrieval of digits from memory. When a digit has been dialled, a Tcl procedure is called that removes the working memory element holding the value of the digit currently being sought and replaces it with the next digit in a string held as a global variable in Tcl. A model of serial recall needs to replace this mechanism.

3 Evaluation of Soar/Tcl-PM

To start to test this model we compared the model's output with the predictions of a similar theory and some informal data. The task we examined was dialing a 11 digit number corresponding to a long distance number in the UK (e.g. 0115 951-5292).

NGOMSL (Kieras, 1998) was used to create two models of dialling a series of digits. The first used a hierarchical decomposition of the task down the lowest level provided for by existing KLM operators -- the operator "move mouse to target" is not decomposed into any sub-methods. This model predicts a time of 23.1 s for dialling a series of 11 digits.

The second similar NGOMS model decomposed the move mouse operator into further subgoals matching the operation of the Soar cognitive model. This extended model gives a predicted timing of 28.6s for dialling a series of 11 digits. For each digit, 13 NGOMSL statements are executed at 0.1 s, and there is 1.3 s of operator time (mouse move 1.1 s + 0.2 s mouse click). This gives 2.6 s per digit, thus 28.6 s for 11 digits.

The different times given by the two NGOMSL models suggest that including perceptual and motor operators at a great level of task decomposition leads to higher predicted timings. However, no data is widely available to support such a level of granularity. Both NGOMS models give predicted timings far in excess of informal times we measured for a real user to dial an 11 digit number, which is approximately 4 to 5 s.

The model was run using four different eye sizes with the memory feature on and with it off. The average times it generated over 10 trials are shown in Table 1. For each trial the eye's initial location was set to be within one saccade right of the '1' button. The model always begins by saccading to the right so this strategy ensured that the '1' button always came into view at the same point in the model's operation. These times are calculated by summing the number of Soar decision cycles executed with the external operator time required. Each Soar decision cycle is assumed to require 0.1 s (Newell, 1990), and the operator time is 1.3 s as in the NGOMSL model. So, for example, a predicted time of 15.4 s is derived from the model requiring 21 decision cycles (2.1 s) in addition to 14.3 s (1.3 s x 11 digits) of external operator time.

Table 1 shows that the memory feature has a marked effect on the model's performance. When the model is able to remember the positions of buttons previously fixated upon, its performance is much more consistent. The size of the eye's perceptual area has relatively little effect on completion of the task. The timings generated in the memory condition are somewhat lower than the timings predicted by either NGOMSL model.

A better match to the NGOMS times is found in the no memory condition, with the model taking between 24 and 30 seconds to complete the task. Perhaps surprisingly, the model's performance does not steadily improve with larger eye sizes. This is most likely due to the way in which the search algorithm is implemented. The eye only reverses its scanning direction once no buttons are visible. In reality, users probably recognise the edge of the keypad much sooner. For larger eye sizes, this results in many unnecessary saccades. This effect is probably negated

In N. Taatgen & J. Aasman (Eds.), *Proceedings of the 3rd International Conference on Cognitive Modelling*. 202-209. Veenendaal, The Netherlands: Universal Press.

Table 1: Results of model trials.

	Time to complete dialling (s)	
Eye Size	Memory on	Memory off
20 x 20	15.4	30.2
50 x 50	15.4	31.5
100 x 100	16.2	32.3
150 x 150	16.2	22.3
200 x 200	16.2	24.3

by the advantages of the larger scanning area of the eye until the eye reaches a size of around 150 x 150, and then the excessive movements required exceed the advantage of the larger view.

4 Discussion and Conclusions

The Soar/Tcl-PM system has become more usable and functional. The interface has been enhanced, many bugs have been eliminated, and the cognitive model has been extended to perform the task of dialling a series of digits with several interfaces not created for use with Soar/Tcl-PM. The original modular structure of the system has been preserved to allow the easy addition of later enhancements to Soar/Tcl-PM's operation and extensions to its cognitive model.

The process of implementing a cognitive model to perform the task of dialling a series of digits in a simulated task domain raises numerous issues about the salience of perception and action in simple goal-oriented tasks. As has been previously noted (Ritter, et al., in press), creating models that can routinely interact with their task environment is difficult enough that the results should be reused. Soar/Tcl-PM is no exception.

4.1 Motor & Perceptual Activity is a Task not an Action

Current tools for the description of task performance, even at low levels of complexity, do not reflect the true nature of the processes and mechanisms necessary for more realistic performance of these tasks. This is due to a lack of data about how people perform at this level, and to the traditional assumption of black-box perceptual and motor processes in cognitive architectures.

Whilst researchers working with cognitive architectures are starting to see that modelling realistic task performance will have to include more detailed perceptual and motor mechanisms, task analysis methods are still limited in the level of task resolution they offer. Motor and perceptual operations are commonly referred to as actions, where an action is a task requiring no control functions. Anyone accepting this definition of motor and perceptual processes has never built a model capable of such activities.

Despite a plethora of different methodologies, techniques, and tools intended to achieve the aims outlined above, the actual notion of task itself is becoming increasingly difficult to pin down. The continuing trend for a shift in HCI towards more social contexts and perspectives has led to the importance of task being played down. However, it is hoped that this project alone should demonstrate that the role of task analysis is ever more important in system design, when it is realised that 'task' itself refers to a wider variety of cognitive and perceptual activities than might previously have been thought.

4.2 The Need for Lower Levels of Analysis

With human-computer interactions being represented as the exchange of states and intents (Briggs, 1988), it is not enough to focus on the existence of artefacts. The processes involved in how a user actually perceives the states of a system are just as important as ensuring that those states are there to be perceived. Computer use has also been likened to problem solving activities, with external objects and situations being translated into internal representations. This

In N. Taatgen & J. Aasman (Eds.), *Proceedings of the 3rd International Conference on Cognitive Modelling*. 202-209. Veenendaal, The Netherlands: Universal Press.

assumption can be traced back as far as 1967 (Craik, 1967) and remains prevalent today. But the study of how we actually perform this translation process is equally if not more salient than studying how we then manipulate our internal representations to arrive at solutions to problems.

4.3 Theories of Vision

There is no shortage of theories of low level perception, particularly vision. However, such theories and the research surrounding them tend to deal with how it is that low-level representations are formed and not with how this knowledge is then used in higher-level cognitive processes. There is even less work dealing with how these higher-level cognitive processes might influence low-level perception. Theories of visual processing tend to deal with static perception and scene recognition, but what is needed is something more like information pick-up theories of Gibson. In a sense, there should be a level of analysis sandwiched between current low-level research and work on cognitive processes. The question of how these two processes interact is an important issue for this field of research.

4.4 Desirable Directions for Extending Soar/Tcl-PM

There are numerous problems remaining with this application and with this approach. Harris (1999) first noted many of these limitations and some of the possibilities for additional capabilities and operation. The ones that still seem particularly applicable include: (a) a lack of knowledge about how real users scan computer displays, (b) a lack of knowledge about how real users co-ordinate mouse movements with such scanning, (c) a lack of comprehensive communication facilities between Soar and Tcl/Tk, (d) a lack of implementation of realistic perceptual moderators such as noise, errors, memory, and learning.

The two most interesting extensions to address these problems are to include search strategies and to include a more realistic model of learning and memory. The search strategy implemented by this project simply refines the default sweeping scan by restricting it to scanning the digit buttons rather than the entire screen. Strategies employing knowledge about button locations are even more desirable. There are three possibilities particularly worth considering.

(a) Inference of button position by number. The number of the digit button currently fixated upon and the number being sought should be compared to given an indication of which direction the eye should move next. On a standard telephone keypad numbers have common positions. If the eye is fixated on '5', and the number being sought is '1', then this knowledge could be applied to cause the eye to start searching up instead of down.

(b) Inference of button position by row knowledge. More specific information about the location of numbers on the keypad could lead to an even more effective scanning strategy. Each row contains only three buttons, so if the digit being sought is outside the range *current digit plus or minus 2* then it is not located on that row. The strategy in the previous strategy would then indicate whether to search up or down.

(c) Knowledge of absolute button positions. The extreme level of knowledge about the telephone interface would be specific knowledge of the absolute location of every digit button. At this level no search would be required since acquired knowledge would specify all necessary information about where to move the eye. This model allows each of these strategies and their combinations to be explored, including how search knowledge is encoded and applied.

A more realistic model of learning and memory is also needed. The processes by which users come to acquire the knowledge to apply the strategies outlined above are not well understood. Extending the model to give it the ability to acquire such knowledge through experience of interfaces would an important development.

As is detailed above, the model remembers the location of each button it fixates upon, allowing it to quickly move the eye to the location of that button should it be sought again later.

In N. Taatgen & J. Aasman (Eds.), *Proceedings of the 3rd International Conference on Cognitive Modelling*. 202-209. Veenendaal, The Netherlands: Universal Press.

The working memory elements used to store this information remain until Soar is re-initialised or quit, and there are no limits on the number of buttons that the model can remember in this way. A more realistic implementation of the memory feature would include limited capacity, errors of storage and retrieval, and decay of stored information over time. Again, data on actual user behaviour in this area is limited, although more general studies of working memory could be applied at least in the first instance.

4.5 Summary

Soar/Tcl-PM has been extended to perform a task using telephone interfaces. This application continues to show the value of being able to tie models to task simulations in this way. The theories of visual perception inherent in implementing Soar/Tcl-PM's search strategy require more human data to allow more psychologically plausible models to be built. The building of these models should then yield increasing support for the relevance of visual search strategies to HCI research and the importance of having models with lower-level implementations of perceptual and motor processes.

References

Altmann, E. M., & John, B. E. (1999). Episodic indexing: A model of memory for attention events. *Cognitive Science, 23(2)*, 117-156.

Anderson, J. R., Matessa, M., & Lebiere, C. (1998). The visual interface. In J.R. Anderson & C. Lebiere (Eds.) *The atomic components of thought*. Mahwah, NJ: Lawrence Erlbaum.

Bass, E. J., Baxter, G. D., & Ritter, F. E. (1995). Creating models to control simulations: A generic approach. *AISB Quarterly, 93*, 18-25.

Briggs, P. (1988). What we know and what we need to know: the user model versus the user's model in HCI. *Behaviour and Information Technology, 7(4)*, 431-442.

Byrne, M. D., & Anderson, J. R. (1998). Perception and action. In J. R. Anderson & C. Lebière (Eds.), *The atomic components of thought.* Mahwah, NJ: Lawrence Erlbaum.

Card, S., Moran, T., & Newell, A. (1983). *The psychology of human computer interaction*. Hillsdale, NJ: Lawrence Erlbaum.

Craik, K.J. (1967). *The nature of explanation*. Cambridge: Cambridge University Press.

Gray, W. D., John, B. E., & Atwood, M. E. (1993). Project Ernestine: Validating a GOMS analysis for predicting and explaining real-world task performance. *Human-Computer Interaction, 8(3)*, 237-309.

Harris, B. (1999). PracTCL: An application for routinely tying cognitive models to interfaces to create interactive cognitive user models. University of Nottingham: Unpublished B.Sc. thesis.

John, B. E., & Kieras, D. E. (1996). Using GOMS for user interface design and evaluation: Which technique? *ACM Transactions on Computer-Human Interaction, 3*(4), 287-319.

Jones, G., Ritter, F. E., & Wood, D. J. (2000). Using a cognitive architecture to examine what develops. *Psychological Science, 11(2)*, 1-8.

Kieras, D. (1998). A guide to GOMS model usability evaluation using NGOMSL. In M. Helander (Ed.) *Handbook of human-computer interaction.* Amsterdam: Elsevier.

Lonsdale, P. R. (1999). Extending PracTCL to provide a more functional eye and hand for the Soar cognitive modelling architecture. MSc thesis, Psychology, U. of Nottingham.

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

Ousterhout, J. (1994). *Tcl and the Tk toolkit*. Reading, MA: Addison-Wesley.

Ritter, F. E., Baxter, G.D., Jones, G., & Young, R.M. (in press) Supporting Cognitive Models as Users. *ACM Transactions on Computer-Human Interaction.*

Zettlemoyer, L. S. & St. Amant, R. (1999). A visual medium for programmatic control of interactive applications. In *Proceedings of the CHI '99*. 199-206. New York, NY: ACM.