Open access • Book Chapter • DOI:10.1007/978-94-010-0421-3_10

A User Modeling Design Tool Based on a Cognitive Architecture for Comparing Interfaces — Source link

Frank E. Ritter, Dirk Van Rooy, Robert St. Amant

Institutions: Pennsylvania State University, North Carolina State University

Published on: 01 Jan 2002

Topics: User interface design, User experience design, Interactive systems engineering, Heuristic evaluation and User modeling

In C. Kolski & J. Vanderdonckt (Eds.), Computer-Aided Design of User Interfaces III, Proceedings of the 4th International Conference on Computer-Aided Design of User Interfaces CADUI'2002. 111-118. Kluwer Academics Publisher, Dordrecht.

A USER MODELING DESIGN TOOL BASED ON A COGNITIVE ARCHITECTURE FOR COMPARING INTERFACES

Frank E. Ritter and Dirk Van Rooy
School of Information Sciences and Technology
The Pennsylvania State University
ritter@ist.psu.edu, dvanrooy@ist.psu.edu

Robert St. Amant
Department of Computer Science
North Carolina State University
stamant@cs.ncsu.edu

Abstract Usability evaluation has long been recognized as an indispensable part of the development of interactive software. Evaluation can still be a difficult and labor-intensive task with conventional tools and user studies, however. Results can vary with the knowledge and experience of the evaluator and with the subjects. We describe an approach to evaluation, based on cognitive modeling, that will alleviate some of these problems. We are building a general-purpose, cross-platform architecture in which user performance in an interactive environment is simulated by a cognitive model. The results of the model can be analyzed and fed back into the development lifecycle. This project is still preliminary, but progress is accelerating.

Keywords: Usability analysis, cognitive modeling

Introduction

Research in automated user interface generation has traditionally concentrated on the specification, design, and implementation stages in the software development lifecycle (e.g. [13].) Over the past several years, however, attention has increasingly focused on the evaluation stage, with tools growing more sophisticated to assist with a greater part of the analysis process. Tools for user interface evaluation vary along several dimensions of effectiveness: at which development stage the technique can be applied; how much knowledge, time, and effort are required in its application; the formal underpinnings of the

technique; the reliability and replicability of its results; and the role of the user in the evaluation. An approach to evaluation that provides novel solutions in several of these areas is outlined in a recent issue of the *International Journal of Human-Computer Studies*, which describes cognitive models that test user interfaces [11]. The approach treats the design of human-computer interfaces as a form of engineering design. Cognitive models provide a means of applying what is known about psychology to predict time, errors, and other measures of user interaction.

If cognitive models are to be used for evaluation, their development and application must be fast and routine. A recent trend is for models to be built within the fixed framework of a cognitive architecture that supports these needs [9]. Increasingly, cognitive architectures are being extended by simulated eyes and hands, enabling the construction of embodied models. Being embodied allows models to interact directly with interfaces. The resulting models can be used to evaluate the interfaces they use and serve as explanations of users' behavior.

There are several advantages to this approach. While we like users (and the authors are themselves users), running usability studies is time consuming and expensive. Performance varies across time and across users, and results are often not reusable. This variability complicates the task of evaluation. Using a cognitive model based on advanced theories of human cognition—that have been validated by user studies—will support a more uniform evaluation, with detailed information collected at a lower cost than with actual users.

Our research has produced simulated eyes and hands for cognitive models that interact with user interfaces [10, 12]. We are now working on ways of making their application more routine. In this paper we present an approach to comparing interface designs through the application of user models based on a cognitive architecture. This approach will generate what we call Cognitive Model Interface Evaluation (CMIE) tools, systems that support the display of the user interface, experimental control over the cognitive model and its simulation runs, feedback on model execution, model execution diagnostics, and simple display facilities for model traces. No such CMIE tools currently exist (though APEX is a step in this direction [4]); however, significant functionality can already be found in each of these areas.

A prototype

As a step towards creating and using a CMIE tool, we have recently started a project targeted at the evaluation of human-robot interfaces. Human-robot interfaces are a challenge that will provide all the types of interface interaction and mental models that we could want.

Figure 1 shows a prototype implementation of the interface to the CMIE tool. This interface has been realized in Visual Basic, and will be translated as we

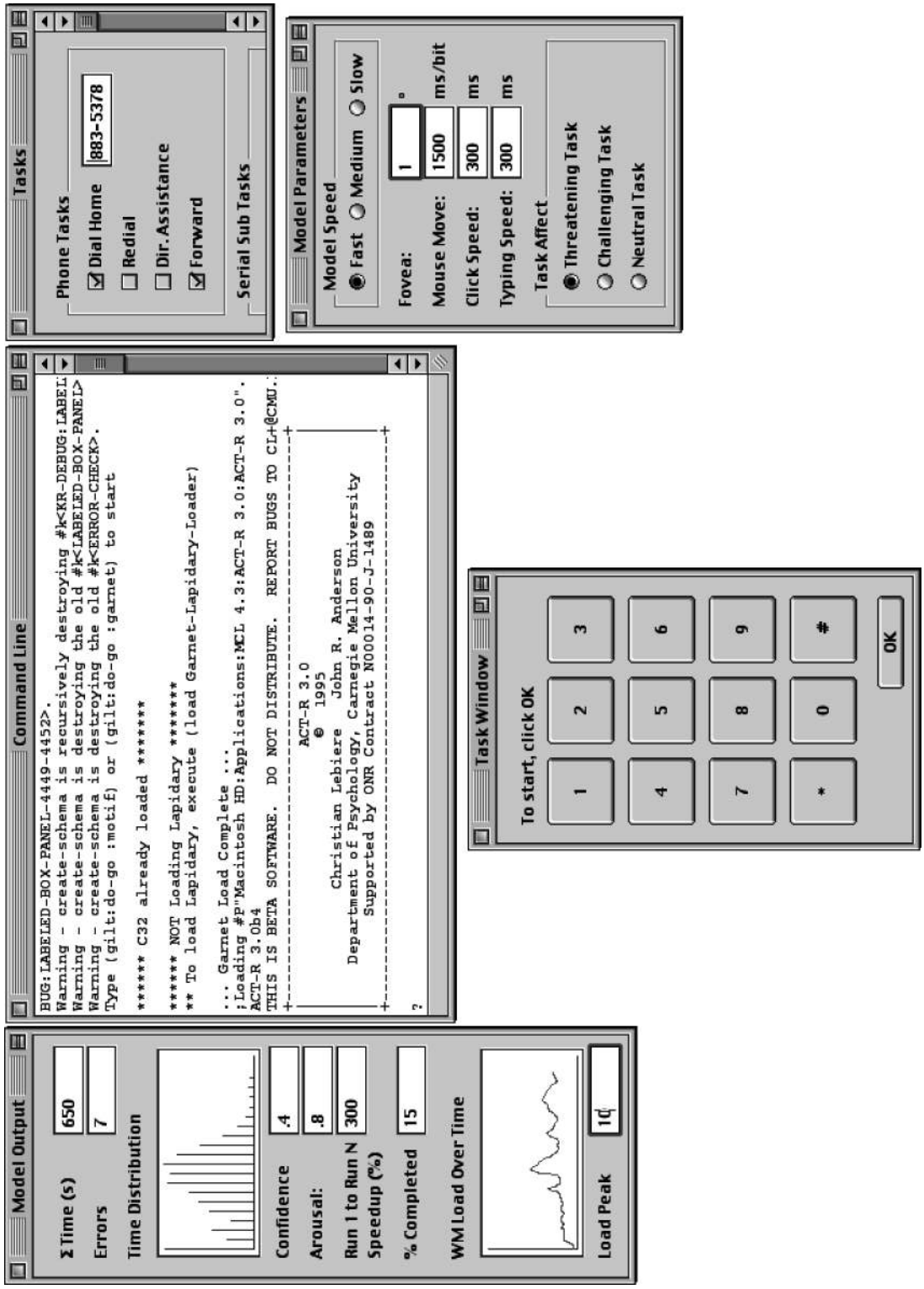

Figure 1. A draft design for a cognitive model-based interface evaluation (CMIE) tool.

note below. Creating a mockup design provides several useful lessons, a way to summarize our progress and the open tasks.

A cognitive modeling architecture. Our first need is to choose a language or theory in which to create our user models. The ACT-R cognitive architecture [1], as seen in the general-purpose Lisp interaction window at the top of the figure, is a suitable candidate. It provides accurate timing predictions, which are important for predicting and producing user strategies. It is used by a wide community of psychologists and user modelers. In time, this will provide us with components to add to our system, including user models, model optimization techniques, and improved versions of the software.

A model of perceptual/motor behavior. A cognitive model needs a way to interact with a user interface. One effective approach is to extend a user interface management system (UIMS) to support cognitive models as users, creating a cognitive model interface management system (CMIMS). A CMIMS manages the interactions of a cognitive model with the interface in the same way that a UIMS manages a real user's interaction with the interface, by managing input and output between the user interface and the model. CMIMSs have been built for several domains, including air traffic control, telephone dialing, and puzzle solving [10], and are now routinely used to tie cognitive models of users to interfaces [2, 8, 10]. ACT-R/PM [1] is an example of a CMIMS. It is suitable because it is integrated with ACT-R, and provides a strong psychology theory on how interaction occurs.

Recently, a potentially more general and robust approach has been developed. It is possible to provide cognitive models access to interfaces based on reading the screen bitmap, parsing it, and passing the results to the model [12]. We will use this approach to provide ACT-R/PM more direct access to the interface, removing the need to create a CMIMS based on a specific UIMS. This will provide a platform- and application-independent architecture for the evaluation of user interfaces based on cognitive user models.

A tasking language. A given cognitive model may be designed to carry out any number of distinct tasks. The tasks window, upper right, allows the modeler to select and refine a specific task, such as dialing a specific number from memory. The decomposition of this task into subtasks and primitive operations can also be under the modeler's control. The figure shows a checklist of tasks for the model to perform, rather than a set of ACT-R production rules. In general, we will provide users with a convenient graphical interface for defining tasks, or a high-level programming language, as was presented by John Anderson at an ACT-R workshop in July, 2001 (http://act.psy.cmu.edu/ACT-R_5.0). Anderson's interpreter takes instructions in a Prolog-type language and creates an ACT-R model that performs the task. This type of interface will be necessary for this approach to be successful.

A theory of individual differences. With different parameter settings, cognitive models can be tailored to reproduce the performance of different classes of users, as shown in the window in the lower right. These settings deal with reaction times (e.g. slow, medium, or fast [3]), properties of the visual system (e.g. the size of the fovea in degrees), and motor performance. We will also take advantage of recent working modeling user's different working memory capacities, which will be an important moderator of behavior and an important measure as well. One class of inputs, task affect, is related to a novel aspect of our cognitive modeling work. In realistic settings, external time constraints and other environmental pressures as well as internal uncertainty can drastically influence performance for even routine tasks. Experimenting with task affect can provide insight into variations in user performance.

Multiple models of user behaviors. We will initially be able to create simple models of menu use and we have in hand models of telephone dialing. This is where we benefit from using the ACT-R cognitive architecture. Many others have created models in ACT-R that are appropriate and useful in a CMIE tool. Recent models include mathematics, graph reading (a list is available at http://act.psy.cmu.edu/papers) These models may not be directly usable, but this project will be a natural consumer of such models, promoting reuse.

Tasks to evaluate. The task window at the bottom of Figure 1 shows the user interface, to be operated by the model. In most cases this will be identical to the interfaces seen by real users. A telephone interface is shown; the user selects a sequence of buttons and then presses OK to finish [6].

Output displays. Cognitive modelers may be well prepared to interpret a model's run, but other users (e.g., software developers) may not. We thus provide more than the model trace. The output window, on the far left, gives a running summary of the model's performance over some number of trials. The window shows general information, such as the number of trials to be carried out and the distribution of trial duration, as well as model-specific information: the running average and peak load on working memory. All of these measures are directly available from ACT-R, and we are displaying such variables in other projects using strip chart displays.

A graphics toolkit. One final component we will need is a language with which to build interfaces for the CMIE tool. We have found that it is most efficient to use the language that the cognitive architecture resides in. The Garnet toolkit [7], while no longer supported, is still quite usable. We are using it in several related, ongoing modeling projects working with ACT-R [5]. Importantly, it is free, and we are able to run it on multiple platforms with multiple version of Lisp (Mac, Unix, and most recently Windows). This multiplatform support is crucial to our claim for the generality of the CMIE approach.

Recent progress

We have taken significant steps towards the realization of the prototype sketched above. For our human-robot interface project, a task has been selected, namely the Mars rover game developed by National Media Technologies (http://www.natlmedia.com/html/fun_mars.html). The type of interface (plan view, controls) and the behavior required from the user (one is on patrol in Mars and the job is to collect alien specimens) are typical of human-robot interfaces. Furthermore, because it is free, it can be modified and run at multiple sites. In later stages, this will be replaced by a more elaborate interface, which may include, for instance, a first person view.

The team at the North Carolina State University, under supervision of St. Amant, has succeeded in letting a simulated eyes and hands module called SEGMAN, interact with the Mars rover game (http://www.csc.ncsu.edu/faculty/stamant/cognitive-modeling.html). At the moment, the module can extract enough information from the scene to allow effective interaction, processing features such as color, position and size. Furthermore, they are establishing software-level integration between Java-based image processing algorithms and Lisp-based modeling software.

Concurrently, Ritter and his collaborators at the Applied Cognitive Science Laboratory are establishing a stable interaction between ACT-R 5.0 (http://act.psy.cmu.edu/ACT-R_5.0) and SEGMAN. Compared to previous implementations, this version of the ACT-R cognitive architecture has qualities that make it even more suitable for projects like ours (more tightly integrated with a perceptual and motor module, event-centered processing allowing interruption). At the moment, the system (ACT-R 5.0 and SEGMAN) runs routinely on Windows machines. A GOMS-like [3] user knowledge structure of the Mars rover game has been generated that can be easily implemented into the production system of ACT-R 5.0, which will allow the model to observe and manipulate objects in the interface. Furthermore, an overlay exists for the ACT-R model that allows modeling the influence of behavioral moderators, such as task appraisal and cafeine (http://ri tter.ist.psu.edu/html/acs-lab#emotions). This overlay will enable the system to simulate individual differences between users, the influence of task affect, and also the influence of substances like nicotine and cafeine on the performance of users. As a result of these joint efforts, a demo of an ACT-R 5.0 model interacting with the Mars rover game is, at this point, clearly within reach.

Summary

This approach to applying cognitive models to design offers a way to summarize and apply what is known in psychology to interface design. It further

promises to make interface evaluation more routine and more accurate, helping to make interface design more like engineering design.

As with all evaluation techniques, the CMIE approach has disadvantages and restrictions. A CMIE evaluation with a cognitive model can be applied only late in the development process, with complete interfaces. The models require empirical validation before they can be applied effectively, and while a CMIE tool decreases the reliance on user studies, considerable knowledge is required to develop appropriate cognitive models.

The novelty of our approach to applying cognitive modeling to evaluate interface designs is that the tool itself is designed for ease of use. It provides generality across platforms and development environments. It makes cognitive modeling tools more readily available, complementing less formal techniques. This approach is not a panacea, and it is not complete. But it is becoming much more within reach, and we, and others, are striving to achieve it.

Acknowledgments

This project is sponsored by the National Science Foundation, award IIS-0083281, and by the Space and Naval Warfare Systems Center, San Diego. The views in this paper do not necessarily reflect the position or the policies of the U.S. Government, and no official endorsement should be inferred.

References

[1] John Anderson and Christian Lebiere. *The Atomic Components of Thought*. Lawrence Erlbaum, Mahwah, NJ, 1998.

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

[3] Stuart K. Card, Thomas P. Moran, and Allen Newell. *The psychology of human-computer interaction*. Lawrence Erlbaum Associates, Hillsdale, NJ, 1983.

[4] Michael A. Freed, Michael G. Shafto, and Roger W. Remington. Using simulation to evaluate designs: The apex approach. In *Proceedings of Engineering for Human-Computer Interaction*, Crete, Greece, 1998.

[5] G. Jones, F. E. Ritter, and D. J. Wood. Using a cognitive architecture to examine what develops. *Psychological Science*, 11(2):93–100, 2000.

[6] P. R. Lonsdale and F. E. Ritter. Soar/Tcl-PM: Extending the Soar architecture to include a widely applicable virtual eye and hand. In N. Taatgen and J. Aasman, editors, *Proceedings of the 3rd International Conference on Cognitive Modelling*, pages 202–209, Veenendaal, NL, 2000. Universal Press.

[7] Brad A. Myers, Dario Giuse, Roger B. Dannenberg, Brad Vander Zanden, David Kosbie, Ed Pervin, Andrew Mickish, , and Philippe Marchal. Garnet: Comprehensive support for graphical, highly-interactive user interfaces. *IEEE Computer*, 23(11), November 1990.

[8] E. Norling and F. E. Ritter. Embodying the JACK agent architecture. Under review.

[9] F. E. Ritter, R. M. Jones, and G. D. Baxter. Reusable models and graphical interfaces: Realising the potential of a unified theory of cognition. In U. Schmid, J. Krems, and F. Wysotzki, editors, *Mind modeling—A cognitive science approach to reasoning, learning and discovery*, pages 83–109. Pabst Scientific Publishing, Lengerich, Germany, 1998.

[10] Frank E. Ritter, Gordon D. Baxter, Gary Jones, and Richard M. Young. Supporting cognitive models as users. *ACM Transactions on Computer-Human Interaction*, 2000.

[11] Frank E. Ritter and Richard M. Young. Embodied models as simulated users: Introduction to this special issue on using cognitive models to improve interface design. *International Journal of Human-Computer Studies*, 55:1–14, 2001.

[12] Robert St. Amant and Mark O. Riedl. A perception/action substrate for cognitive modeling in HCI. *International Journal of Human-Computer Studies*, 55(1), 2001.

[13] Joseph W. Sullivan and Sherman W. Tyler, editors. *Intelligent User Interfaces*. ACM Press, 1991.