To appear in *BRIMS 2005*. Draft of 9 April 2005.

A Tutorial on Herbal: A High-Level Language and Development Environment Based on Protégé for Developing Cognitive Models in Soar

Frank E. Ritter, Geoffrey P. Morgan, William E. Stevenson
Applied Cognitive Science Lab (acs.ist.psu.edu)
School of Information Sciences and Technology
The Pennsylvania State University
University Park, PA 16802
frank.ritter@ist.psu.edu, gmorgan@psu.edu, wstevenson@ist.psu.edu

Mark A. Cohen
Business Administration, Computer Science, and Information Technology
Lock Haven University
Lock Haven, PA 17745
mcohen@lhup.edu

Keywords: Cognitive Modeling, Development Environments, High-Level Languages, Ontologies, Protégé, Soar

ABSTRACT: *To promote the use of cognitive systems, it is essential that tools such as high-level languages and development environments are created to allow the modeler to focus more on the problem domain, and less on the nuances of a particular architecture. This tutorial introduces a development environment and high-level behavior representation language called Herbal that represents a step towards creating tools to support a wide range of cognitive model users. The design of Herbal is based on a study of what questions users ask of a model (Haynes, Councill, & Ritter, 2004). It appears to offer a faster way to program Soar.*

1. Introduction

Designing, implementing, and using cognitive architectures can be a difficult task considering that the background and expertise of this diverse set of users varies from novice to expert (Ritter et al., 2003). Part of the problem stems from the fact that cognitive architectures typically use low-level programming languages to model behavior. Both Soar (Laird, Congdon, & Coulter, 1999) and ACT-R (ACT-R Research Group, 2004), for example, use production rules as their primary programming construct.

A wider variety of users can be supported in the task of developing cognitive models using a language that maps more directly to the domain that the user is familiar with. For example, if this high level language is built on top of Soar, and closely resembles the theory used by the Soar architecture (Newell, 1990), it would make Soar easier to use, while maintaining the features of Soar that make it interesting, including learning, interaction, and interuptability.

Several tools already exist to simplify the development of cognitive architectures. For example, Soar developers have VisualSoar (Laird, 1999), a powerful development environment that simplifies the creation of Soar productions. SoarDoc (Soar Technology Inc., 2004) is a tool for automatically generating HTML based documentation directly from Soar source code. ViSoar (Hirst, 1999) is a dialogue driven interface for generating Soar code automatically, and debugging, and reverse engineering existing Soar productions. For ACT-R there is G2A (St. Amant, Freed, & Ritter, 2005), and other architectures include integrated development environments (IDEs) (Busetta, Rönnquist, Hodgson, & Lucas, 1999; Lebiere et al., 2002; Zachary, Jones, & Taylor, 2002)

This tutorial presents a cognitive modeling tool that involves the use of a high-level programming language and a compiler. This tool is different because it is designed not only to simplify the development of cognitive models, but also to generate explanations of the running models.

1.1 Herbal

This three-hour tutorial introduces an integrated development environment called Herbal (acs.ist.psu.edu/projects/herbal) that provides a way to write Soar models on a higher level than simply writing Soar rules (Cohen, Ritter, & Haynes, 2005). With it, users can create models graphically and have these models compiled into Soar productions. An example screen shot of the Herbal Blocks-World model is shown in Figure 1. The resulting productions are a Soar model. The structure of the model can be passed to an associated tool that can help display and explain the model, even as the model runs (the Herbal Viewer). The design of Herbal is based on a study of what questions users ask of a model (Haynes, Ritter, Councill, & Cohen, 2005).

A tutorial on Herbal is available as a tech report (Cohen & Ritter, 2004). This tutorial is a condensed version of that tutorial with additional, hands-on help.

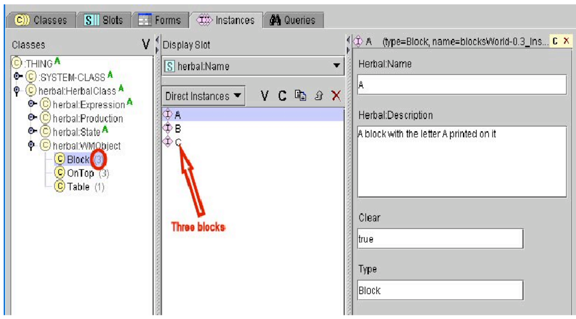

Figure 1. Example screen of the Herbal IDE.

1.2 Intended audience

This Herbal tutorial is of primary interest to those who wish to learn about Soar (Newell, 1990; Laird, Newell, & Rosenbloom, 1987) and about high level languages for modeling cognition. It will also be interesting to those interested in learning more about cognitive modeling with cognitive architectures (Ritter, 2004).

An inadvertent aspect of this tutorial is that tutees will learn about how to use the Protégé ontology editor from Stanford, which has multiple uses as well, such as editing the DAML+OIL ontology. The introduction to dTank (Morgan, Ritter, Stevenson, Schenck, & Cohen, 2005) in the last third of the tutorial helps explain many of the concepts of cognitive modeling, and may be of interest to many in this community as a simple testbed for developing intelligent agents.

2. Tutorial contents

The purpose of this tutorial is not to provide users with complete knowledge of Soar, Protege, and Herbal. It illustrates these tools with a balance of lecture and of hands-on exercises. The exercises will be started but not necessarily completed by the tutees, following the approach of the successful Psychological Soar Tutorial. The exercises will, however, provide the tutees with the initial knowledge necessary for getting started with these tools.

2.1 Introduction to Soar, Herbal, and basic editor (Protégé)

In the first hour, as an introduction we will provide a short overview of Soar and provide the rationale for high-level cognitive modeling and behavior representation languages. With this in hand, we will provide an overview of the Herbal system and its capabilities, and provide a hands-on tutorial of the Protégé ontology editor.

2.2 Example simple models in Herbal

We introduce Herbal by providing two example models for tutees to use in a tutorial lecture, demonstration, exercise format. We will use a blocks world model to start to understand and apply Herbal, and a simple learning model to explore learning in Soar.

2.3 More sophisticated model in Herbal

We introduce the dTank environment (acs.ist.psu.edu/dTank), a microworld designed to have many of the important aspects of ModSAF (e.g., maps, interaction, real-time competitive environment), but far simpler. An example model that interacts with the dTank simulation will be explained, and the tutees will modify the model to be more intelligent.

2.4 Installing the Required Software

Herbal consists of several pieces of software that are available on the net. They work on PCs and Macintoshes. Tutees can install the software by getting instructions from acs.ist.psu.edu/herbal, but support will be provided to install on the day. Tutees will be encouraged to work in teams, so those without laptops will be paired with those with laptops. In the last five years of the Cognitive Science Conference, this sharing has proved a very workable and valuable solution.

3. Brief Overview of the Herbal Development Environment

The Herbal development environment has two main responsibilities: to make it easy to generate a Herbal program, and to translate this program into Soar. The first responsibility is realized using the Protégé integrated knowledge-based development tool (Stanford Medical Informatics, 2004). Protégé makes it possible to graphically define an ontology, create an instance of that ontology, and save it. By combining custom Protégé plug-ins, and the Herbal high-level language ontology, Protégé was configured to create a graphical programming environment for Soar models. Modelers use Protégé to instantiate a model by extending the Herbal ontology, and then saving this model in RDF format. Portions of existing models can be reused by importing existing model ontologies into Protégé and adapting them.

The second responsibility of the Herbal development environment is compiling the generated RDF into productions. XSLT proved to be an excellent solution for compiling the high-level language.

Currently, Herbal generates Soar productions. However, this design makes it easy to support additional target architectures: to support a new cognitive architecture, a new XSLT script must be written and plugged into Herbal.

4. Experiences Using Herbal

Herbal has been used by about 45 users to create models, primarily those that interact with a simple microworld (dTank). Its largest use was in an advanced undergraduate course in Fall 2004, to 38 undergraduates and 3 graduate students. Students provided positive feedback in their formal course evaluations on both Protégé and Herbal as modeling environments. In general, the students encountered fewer problems programming in Soar when using Herbal. This is admittedly an anecdotal result, and could also be the result of other factors including increasing motivation by including more applications to dTank, and referring to the relevance to video games quite often.

Among the advantages noted by students was Herbal's ability to reuse conditions and actions across several operators. This enabled teams to build more complex tanks using a set of predefined conditions and actions specific to the dTank environment. Students also commented on the usefulness of Herbal's ability to automatically generate comments.

Students noted the lack of impasses as one of the major disadvantages (they have been added to the current version 0.9). Another disadvantage noted by the students was related to usability issues with Protégé. A new version of Protégé (version 3.0) has since been released that has addressed many of these issues.

Since then, a single subject analysis of how long it takes to create a dTank model using Herbal (Morgan, Cohen, Haynes, & Ritter, in press) showed that Herbal can be as fast at TAQL (Yost, 1993).

5. Conclusions

The ability to create cognitive models using powerful architectures such as Soar can be simplified using high-level languages. The Herbal high-level language is an example of such a solution and is capable of producing Soar productions from models created using the Herbal development environment, which is based on a modified ontology editor .

Future development of Herbal is currently focused on using the information contained in the Herbal ontology to generate explanations while the model is running. It is believed that runtime explanations such as these will further aid developers, as well as other users, with the creation of cognitive models.

6. Acknowledgements

The development of this software was supported by the Office of Naval Research under Contract N00014-02-1-0021. Herbal was developed with Steve Haynes and Isaac Council. Discussions with Randy Jones, Robert St. Amant, Glenn Taylor, and Bob Wray have informed this work. Urmila Kukreja has helped test Herbal. We also thank the students of IST 402: Emerging Technologies: Models of human behavior the last two years for their work with and comments on Herbal.

7. References

ACT-R Research Group, CMU. (2004). ACT-R 5.0 Tutorial, from act-r.psy.cmu.edu/tutorials/

Busetta, P., Rönnquist, R., Hodgson, A., & Lucas, A. (1999). JACK intelligent agents - Components for intelligent agents in JAVA. *AgentLink News Letter, 2*(Jan.), www.agent-software.com/white-paper.pdf.

Cohen, M. A., Ritter, F. E., & Haynes, S. R. (2005). Herbal: A high-level language and development environment for developing cognitive models in Soar. In L. Allender & T. Kelley (Eds.), *Proceedings of the 14th Conference on Behavior Representation in Modeling and Simulation.* Orlando, FL: U. of Central Florida.

Cohen, M., & Ritter, F. E. (2004). Herbal tutorial (Tech. Report No. 2004-2). Applied Cognitive Science Lab, School of Information Sciences and Technology, Penn State. acs.ist.psu.edu/acs-lab/reports/cohenR04.pdf.

Haynes, S. R., Councill, I. G., & Ritter, F. E. (2004). Responsibility-driven explanation engineering for cognitive models. In R. Jones, R. E. Wray, & M. Scheutz (Eds.), *AAAI Workshop on intelligent agent architectures: Combining the strengths of software engineering and cognitive systems*. 46-52. Menlo Park, CA: AAAI Press.

Hirst, T. (1999). ViSoar - Towards an Agent Development Environment for the Soar Architecture. Paper presented at the *4th Online Workshop on Soft Computing (WSC4).*

Laird, J. E. (1999). Visual Soar. Paper presented at the *Soar Workshop 19*, University of Michigan.

Laird, J. E., Newell, A., & Rosenbloom, P. S. (1987). *Soar: An architecture for general intelligence. Artificial Intelligence, 33*(1), 1-64.

Laird, J. E., Congdon, C. B., & Coulter, K. J. (1999). *The Soar User's Manual Version 8.2*: University of Michigan.

Lebiere, C., Biefeld, E., Archer, R., Archer, S., Allender, L., & Kelley, T. (2002). IMPRINT/ ACT-R: Integration of a task network modeling architecture with a cognitive architecture and its application to human error modeling. Paper presented at the *Advanced Technologies Simulation Conference*, San Diego, CA.

Morgan, G. P., Cohen, M. A., Haynes, S. R., & Ritter, F. E. (2005). Increasing efficiency of the development of user models. To appear in E. J. Bass (Ed.), *Proceedings of the IEEE System Information and Engineering Design Symposium*.

Morgan, G. P., Ritter, F. E., Stevenson, W. E., Schenck, I. N., & Cohen, M. A. (2005). dTank: An environment for architectural comparisons of competitive agents. In L. Allender & T. Kelley (Eds.), *Proceedings of the 14th Conference on Behavior Representation in Modeling and Simulation*. Orlando, FL: U. of Central Florida.

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

Ritter, F. E. (2004). Choosing and getting started with a cognitive architecture to test and use human-machine interfaces. *MMI-Interaktiv-Journal, 7*, 17-37. useworld.net/mmiij/musimms.

Ritter, F. E., Shadbolt, N. R., Elliman, D., Young, R., Gobet, F., & Baxter, G. D. (2003). *Techniques for modeling human and organization behavior in synthetic environments: A supplementary review,* Wright Patterson Air Force Base, OH: Human Systems Information Analysis Center.

St. Amant R., Freed A. R., & Ritter, F. E. (2005). Specifying ACT-R models of user interaction with a GOMS language. *Cognitive Systems Research, 6*(1), 71-88.

Soar Technology Inc. (2004). SoarDoc, from www. eecs.umich.edu/~soar/sitemaker/projects/soardoc/ soardoc.html

Stanford Medical Informatics. (2004). Protege (Version 2.1.1). protege.stanford.edu

Yost, G. R. (1993). Acquiring knowledge in Soar. *IEEE Expert, 8*(3), 26-34.

Zachary, W., Jones, R. M., & Taylor, G. (2002). How to communicate to users what is inside a cognitive model. Paper presented at the *11th Computer Generated Forces Conference,* Orlando, FL.

The Tutorial Team

FRANK RITTER (www.frankritter.com) is in The School of Information Sciences and Technology, and is also affiliated with Psychology and with Computer Science, at The Pennsylvania State University. He works on the development, application, and methodology of cognitive models, particularly as applied to interfaces and emotions. Ritter helped start the International Conference on Cognitive Modeling. With Richard Young he started a tutorial series at the Cognitive Science Conference and was co-chair for five years. With Richard and others he has given the Psychological Soar tutorial over 15 times on three continents at conferences, as staff development, and as part of graduate and undergraduate courses. He wrote the original Soar FAQ, and a new release is expected this spring.

He is on the editorial board of *Human Factors* and *AISB Journal*. His review (with others) on applying models in synthetic environments was published as a HSIAC State of the Art Report (iac.dtic.mil/hsiac/S-docs/ SOAR-Jun03.pdf). Ritter earned his PhD in AI and Psychology and a MS in psychology from Carnegie-Mellon, and a BSEE from UIUC.

MARK COHEN is an instructor in the Business Administration, CS and IT Department at Lock Haven University, and a PhD student in the School of IST at Penn State. His current research efforts include developing software that simplifies the creation and maintenance of cognitive models—he is the current developer of Herbal. He received an MS in CS from Drexel University and a BS EE from Lafayette College. He has over 10 years of experience developing health care and pharmaceutical software.

GEOFFREY P. MORGAN is a student in the School of IST at Penn State. His current work focuses on modeling behavioral moderators and studying the actions of teams within contextual units and as coordinated individuals. He has used several versions of Herbal.

WILLIAM STEVENSON is a student in the School of IST at Penn State. His current work focuses on modeling human vision and creating displays for explaining cognitive models, and he has helped teach Herbal at Penn State. He is the current editor in chief of *ACM Crossroads*. Stevenson received a BA/BS dual degree in Cognitive Science and Computer Science from Muhlenberg College in Allentown, PA and a MEng in Computer Science and Engineering from Penn State.