Prologue - Making Models Easy to Use

- Elkind, Card, Hochberg, & Huey. (1988) - We need work

- Pew & Mavor
  books.nap.edu/catalog/6173.html
  - Ready to use

  iac.dtic.mil/hsiac/S-docs/SOAR-Jun03.pdf
  - Usability also counts
  - Lists of open tasks
Usability/Affordability has Many Facets

- What to do and
- How to do it


- Better languages
- Better training (e.g., tutorials, Tank-Soar)

Making models explainable - What we do

- What users need? **
- Software techniques to create the tool **
- Tools based on user needs and theories of explanation and HCI **
- Allow models to access interfaces **
- Allow modelers access to models (SPICE/FSF)**
Making Soar More Articulate and More Understandable

Frank E. Ritter and Steven R. Haynes

Applied Cognitive Science Lab
School of Information Sciences and Technology
Penn State

acs.ist.psu.edu/papers  ritter@ist.psu.edu

Working with Soar Tech for CaDaDis windows, Mark Cohen (Lock Haven U.), Ian Schenck, Kevin Tor, and Robert St. Amant (NCSU)

This project is supported by the US Office of Naval Research, award number N00014-02-1-0021
Talk presented at ONR Workshop, 22 July 2004
What Users Want (Herbal 0a)
(based on 4 expert SAP users, Councill et al., 2003)

Summary: Better description of procedural knowledge needed
Result of Analysis - *Operational Questions*  
(Haynes, Councill & Ritter, in prep.)

<table>
<thead>
<tr>
<th>Question Type</th>
<th>Count (N = 207)</th>
<th>%</th>
</tr>
</thead>
<tbody>
<tr>
<td>What</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Identity</td>
<td>41</td>
<td>20%</td>
</tr>
<tr>
<td>Definition</td>
<td>79</td>
<td>38%</td>
</tr>
<tr>
<td>Relation</td>
<td>6</td>
<td>3%</td>
</tr>
<tr>
<td>Event (what happened?)</td>
<td>6</td>
<td>3%</td>
</tr>
<tr>
<td>How</td>
<td></td>
<td></td>
</tr>
<tr>
<td>How do I use it?</td>
<td>25</td>
<td>12%</td>
</tr>
<tr>
<td>How does it work?</td>
<td>44</td>
<td>21%</td>
</tr>
<tr>
<td>Where</td>
<td>5</td>
<td>2.5%</td>
</tr>
<tr>
<td>When</td>
<td>1</td>
<td>0.5%</td>
</tr>
</tbody>
</table>
# Result of Analysis - Why Questions

<table>
<thead>
<tr>
<th>Question Type</th>
<th>Count (N = 29)</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Functional Explanation</td>
<td>15</td>
<td>52%</td>
</tr>
<tr>
<td>Pragmatic Explanation (contrast classes)</td>
<td>9</td>
<td>31%</td>
</tr>
<tr>
<td>Unclassified</td>
<td>5</td>
<td>17%</td>
</tr>
</tbody>
</table>
Done: Analysis of Soar Explanation Elements

- Deconstruction of Soar architecture to identify explanatory elements
  [Done Aug 03]

- Design-based analysis of CGF explanation-seeking questions
  [Haynes, Soar 23, 12 experts x 1 hours]
  (Haynes et al., 2004, summary to AAAI)
Using a High Level Behavioral Representation Language (Herbal 0b)

- Design rationale anchors model explanations (Haynes, 2001; Haynes et al., 2004)
  - Design knowledge capture during development
- Model description needed for explanations (Haynes, 2003)
  - Create model parts within IDE
- Responsibility-driven approach through a compiler
  - Organize knowledge and rules (Haynes et al., 2004; St. Amant & Ritter, 2004, www4.ncsu.edu/~stamant/G2A)
Herbal Design (0b)

• Augment existing planning language with design rationale
  ✧ Chose RDF and Protoge: tool availability, generality
  ✧ Also studied direct translation (St. Amant & Ritter, 2004)

• Explanation from declarative representation + rationale

• Compile into Soar rules
  ⇒ (could also compile into ACT-R, JACK?)

• Designed to leverage VISTA
  ⇒ (declarative representation supports model tracing)
  acs.ist.psu.edu/vista for our local training examples
HLBRL
- Procedure Descriptions
- Declarative Representations
- Design Rationale

Output:
- Task
- Pert Chart (CPM-GOMS)
- ACT-R Rules
- Declarative Memory
- Operators
- State Augmentations
- Explanation Knowledge

Eye-hand

dTank Sim.

Audio

Display

Users
Herbal IDE (1): Language Overview

Architecture

- Ontology Editor (Protégé)
- Preprocessor
- XSLT
- Soar Rules

Features

- Captures Design Documentation
- Namespacing!
- Support for Global Knowledge Bases
- Support for Importing Domain Ontologies and Model Extensions
- Graphical State Layout
- Exists, alpha
Herbal Integrated Development Environment (IDE)
The purpose of this model is to arrange three blocks on a table in a specified order.
Herbal IDE (2003):
Example Compiler Output

STATE top-state
OPERATOR attack
  PRECOND INPUT ^foodog
  PRECOND <dog1> ^visible
  OUTPUT attack <x1> <y1>
OPERATOR move
  PRECOND INPUT ^location <loc>
  PRECOND <loc> ^<direction>.content empty
  OUTPUT move <direction>

Output ACTION attack x,y
  EFFECT ^x <x1> ^y <y1>

Output ACTION move <direction>
  EFFECT ^direction <dir>

PREFERENCE attack move
CHOICE attack ^il.status.health >= 10
  CHOICE move ^il.status.health < 10

sp {apply*ol*attack
  (state <s> ^operator <o>
   ^io.output-link <ol>)
  (o ^name attack
   ^x <xval>
   ^y <yval>)
  -->
  (ol ^attack <action>)
  (action ^x <xval> ^y <yval>)}

sp {apply*ol*move
  (state <s> ^operator <o>
   ^io.output-link <ol>)
  (o ^name move
   ^direction <dir>)
  -->
  (ol ^move <action>)
  (action ^direction <dir>)}

sp {select*attack*move1
  (state <s> ^operator <o1> +
   ^operator <o2> +
   ^io.input-link.status.mana < 10)
  (o1 ^name attack)
  (o2 ^name move)
  -->
  (s ^operator <o1> > <o2>)}

sp {select*attack*move2
  (state <s> ^operator <o1> +
   ^operator <o2> +
   ^io.input-link.status.mana < 10)
  (o1 ^name move)
  (o2 ^name attack)
  -->
  (s ^operator <o1> > <o2>)}
Herbal IDE 2004: Compiler Output (~ 4x more information)
Herbal Viewer (2)

<table>
<thead>
<tr>
<th>Model</th>
<th>Transition Diagram</th>
</tr>
</thead>
<tbody>
<tr>
<td>top-state (4%)</td>
<td></td>
</tr>
<tr>
<td>wonder</td>
<td></td>
</tr>
<tr>
<td>move (1%)</td>
<td></td>
</tr>
<tr>
<td>turn</td>
<td></td>
</tr>
<tr>
<td>chase</td>
<td></td>
</tr>
<tr>
<td>move</td>
<td></td>
</tr>
<tr>
<td>attack</td>
<td></td>
</tr>
<tr>
<td>attack,attack (16%)</td>
<td></td>
</tr>
<tr>
<td>move</td>
<td></td>
</tr>
<tr>
<td>fire-missile</td>
<td></td>
</tr>
<tr>
<td>retreat</td>
<td></td>
</tr>
<tr>
<td>retreat,retreat (38%)</td>
<td></td>
</tr>
<tr>
<td>move (36%)</td>
<td></td>
</tr>
<tr>
<td>retreat</td>
<td></td>
</tr>
<tr>
<td>move</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>State Trace</th>
<th>Operator Trace</th>
<th>Commentary</th>
<th>Vista Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td>wait (1704)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>move (23)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>wander (7)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>turn (7)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>retreat (7)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>fire-missile (3)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>move (4)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>chase (2)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>attack (2)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>turn (1)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>move (1)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>retreat (1)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Herbal Viewer with Blocks World
Herbal Viewer with TankSoar
Herbal (3): VISTA Display Designed for Sequence Representation, CaDaDis, integration
acs.ist.psu.edu/CaDaDis, act-r, soar, jess, + jack, cast

In progress
## Support in Herbal for Explanation

<table>
<thead>
<tr>
<th>Question Type</th>
<th>Support (partial)</th>
</tr>
</thead>
<tbody>
<tr>
<td>What</td>
<td></td>
</tr>
<tr>
<td>Identity</td>
<td>Viewer IDE</td>
</tr>
<tr>
<td>Definition</td>
<td>planned IDE</td>
</tr>
<tr>
<td>Relation</td>
<td>Viewer IDE</td>
</tr>
<tr>
<td>Event (what happened?)</td>
<td>Viewer</td>
</tr>
<tr>
<td>How</td>
<td></td>
</tr>
<tr>
<td>How do I use it? - tool</td>
<td>Viewer IDE</td>
</tr>
<tr>
<td>- model</td>
<td>planned IDE</td>
</tr>
<tr>
<td>How does it work?</td>
<td>Viewer IDE</td>
</tr>
<tr>
<td>Where</td>
<td>Viewer IDE</td>
</tr>
<tr>
<td>When</td>
<td>Viewer</td>
</tr>
</tbody>
</table>

### Functional Explanation
- Support: planned IDE

### Pragmatic Explanation (contrast classes)
- Support: TBD
Support in Herbal Viewer for Explanation

Planned
How do I use it?
- model
How does it work?
Contrast classes
Manual
How do I use it?

What - structure
What - identity
Definition - structure + IDE
When - order
- how often

What - relation
Herbal (4): Models Interact with Interface Directly

(\cite{Ritter, Baxter, Jones, & Young, 2000})

- Sim-eyes and -hands interact directly with interfaces \((w/St. Amant)\)
  \cite{Shah, St. Amant et al., 2003} \cite{(St. Amant et al., 2004)}

- Avoids instrumenting interfaces
- Needs support in Herbal compiler

Perhaps 20x speedup on phones now?
Herbal (5): dTank Microworld

acs.ist.psu.edu/dTank

- For testing explanation of dynamic, adversarial models
- Multiple players and teams on multiple machines
- Improved 2004: complexity, vision theory, speed, interface, architecture use: 4 (cast, soar, jess, java) + 3 (act-r, jack, CoJACK**)
- Java
- Used by an Army MURI (Sun et al., 2004), at Lock Haven U. (www.lhup.edu/~mcohen/dTank/dTankJess.htm), Federal U. of Uberlandia (Brazil)
Herbal (1b,2b,5b): Users to Use, Test, Expand Herbal & dTank

- IST 402: Models of human behaviour
- Microworld to understand, create, and exercise adversarial Soar models
- Last year, class projects
  - MacSoar compilation
  - Architectural comparisons (Sun et al, 2004)
- Will explore usability aids, how to explain behavior, and when to interrupt users to tell them about their models
Further Benefits

• Help maintainability

• Ease programming
  • 3x Productivity (Yost, 3 min./production)
  • 1x Productivity (IST 402 class, no complaints of Soar)
  • 1x-2x (Kukreja, first two programs @ 10 hours/program)
  • 5-10-30x (St. Amant, reanalysis of ACT-R model, another phone)

• Promote reuse (document and import models)

• Explanations may increase learning and use (not mentioned by our subjects)
Why Will This One Work?

- Principled design based on a theory of knowledge (PSCM, roughly and extended)
- Based on theory of explanations
- Based on data on explanations study
- New payoff - explanations
- Software engineering principles
  - Modularity
  - Software reuse
  - Design patterns
- No lost expressiveness (not yet, but coming)
- Extendable by users / Common tools
- User base lined up for feedback (~12+2+40)
- Designed for usability by CS/HCI/Psy team
More Articulate and More Understandable Soar

- High-level language IDE designed to support explanation from declarative representation + rationale
  - With multi-media delivery
  - Improved developer productivity and wider range of outputs
  - Access to any interface
- Microworld for exploring these issues
  - Audience of users available
  - Architectural and process comparisons are possible
References (acs.ist.psu.edu/papers/)


