Making Soar More Articulate and More Understandable

Frank E. Ritter, Steven R. Haynes, Isaac G. Councill

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

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

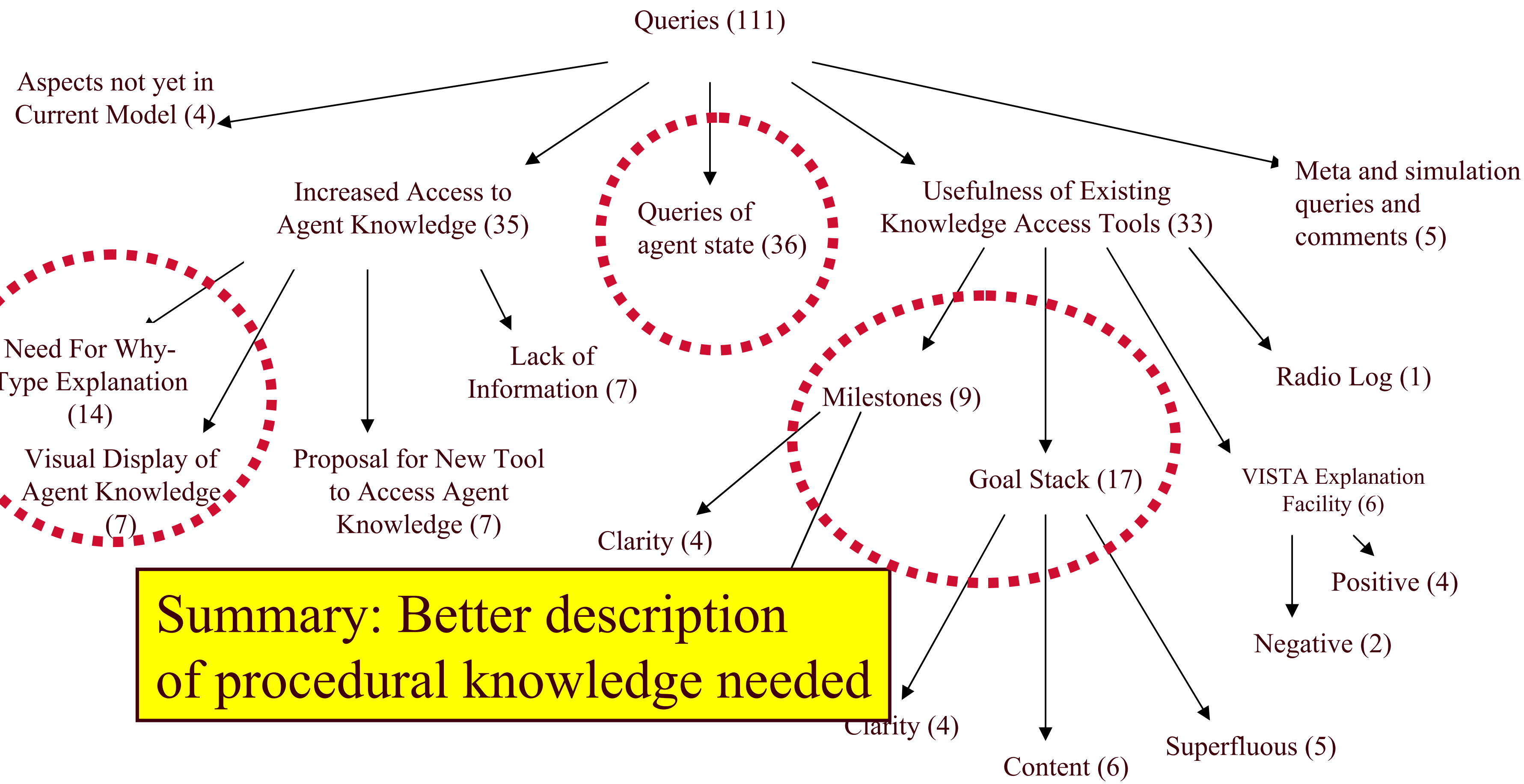
(Working with Soar Tech as needed, as well as with Mark Cohen (Lockhaven U.), Kevin Tor, Alex Wood, and David Mudgett)

This project is supported by the US Office of Naval Research, award number N00014-02-1-0021

Talk presented at Soar 23 Workshop, 26 June 2003

What Users Want

(based on 4 expert SAP users, 🕮 Councill et al., 2003)



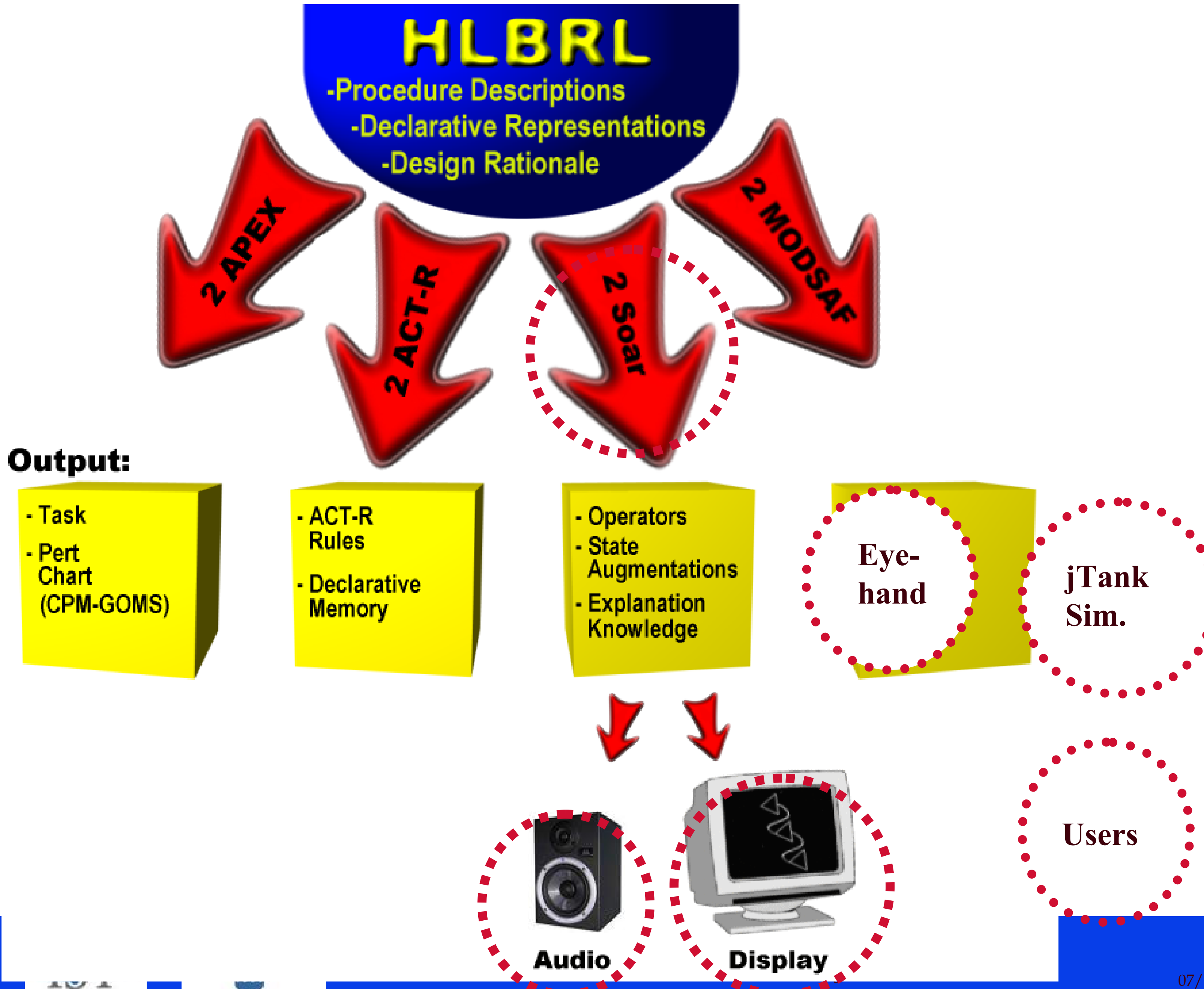
Summary: Better description of procedural knowledge needed

Why a High Level Behavioral Representation Language? (HLBRL)

- Provides model description needed for explanations (📖 Haynes, 2003, here)
- Design rationale anchors model explanations (📖 Haynes, 2002)
- Captured at development time
- Ad'l advantages:
  - Model clarity
  - Supports reuse
  - 3x Dev. Productivity
  - Extensibility

Key Result: Role of High-level Behavior Representation Language

- Augment existing planning language with design rationale
  - Examined 6 candidate languages
  - Chose RDF : tool availability, generality
- Explanation from declarative representation + rationale
- Compile into Soar rules (Allsopp 03a,b, Yost 90)
  - (could also compile into ACT-R, JACK, APEX?)
- Designed to work with VISTA
  - (declarative representation supports model tracing)


HLBRL
-Procedure Descriptions
-Declarative Representations
-Design Rationale
2 APEX
2 ACT-R
2 Soar
2 MODSAF
Output:
- Task
- Pert Chart (CPM-GOMS)
- ACT-R Rules
- Declarative Memory
- Operators
- State Augmentations
- Explanation Knowledge
Eye-hand
jTank Sim.
Audio
Display
Users

HLBRL (Part 1a): Example HLBRL Compiler

```
STATE top-state
  OPERATOR attack
    PRECOND INPUT ^food
    PRECOND <dog1> ^visi
    OUTPUT attack <x1> <y

  OPERATOR move
    PRECOND INPUT ^loca
                     -^move-b
    PRECOND <loc> ^<dire
    OUTPUT move <directio

OutputACTION attack x1 y
  EFFECT ^x <x1>  ^y <y1>

OutputACTION move dir
  EFFECT ^direction <dir>

PREFERENCE attack mov
  CHOICE attack ^il.status.
  CHOICE move ^il.status.h
```

```
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)
   -->
   (<s> ^operator <o1> > <o2>)}
```

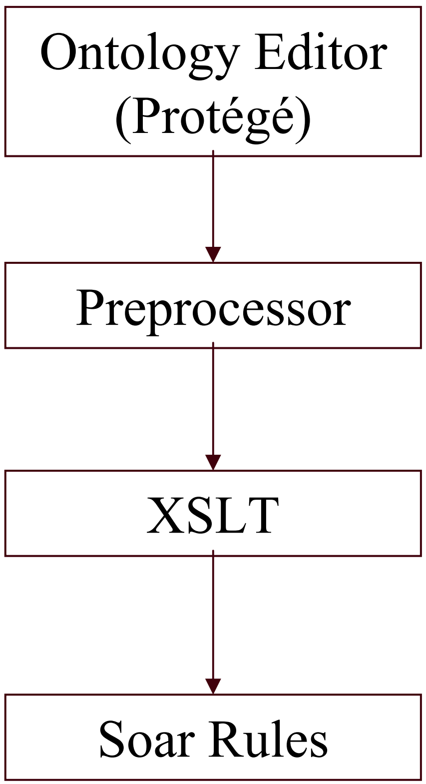
```
sp {select*attack*move2
   (state <s> ^operator <o1> +
              ^operator <o2> +
              ^io.input-link.status.mana < 10)
   (<o1> ^name move)
   (<o2> ^name attack)
   -->
   (<s> ^operator <o1> > <o2>)}

sp {propose*attack
   (state <s> ^superstate nil)
   (<s> ^io.input-link <il>)
   (<il> ^foodog <foodog> ^refreshed <ref>)
   (<foodog> ^visible yes ^x <x> ^y <y>)
   -->
   (<s> ^operator <o> + =)
   (<o> ^name attack)
   (<o> ^x <x> ^y <y> ) }

sp {propose*move
   (state <s> ^superstate nil)
   (<s> ^io.input-link <il>)
   (<il> ^location <loc> -^move-blocked yes ^refreshed
<ref>)
   (<loc> ^<dir>.content empty)
   -->
   (<s> ^operator <o> + =)
   (<o> ^name move)
   (<o> ^direction <dir> ) }
```

acs.ist.psu.edu/articulate/compiler/

IST PENN

HLBRL (Part 1b): Language Overview

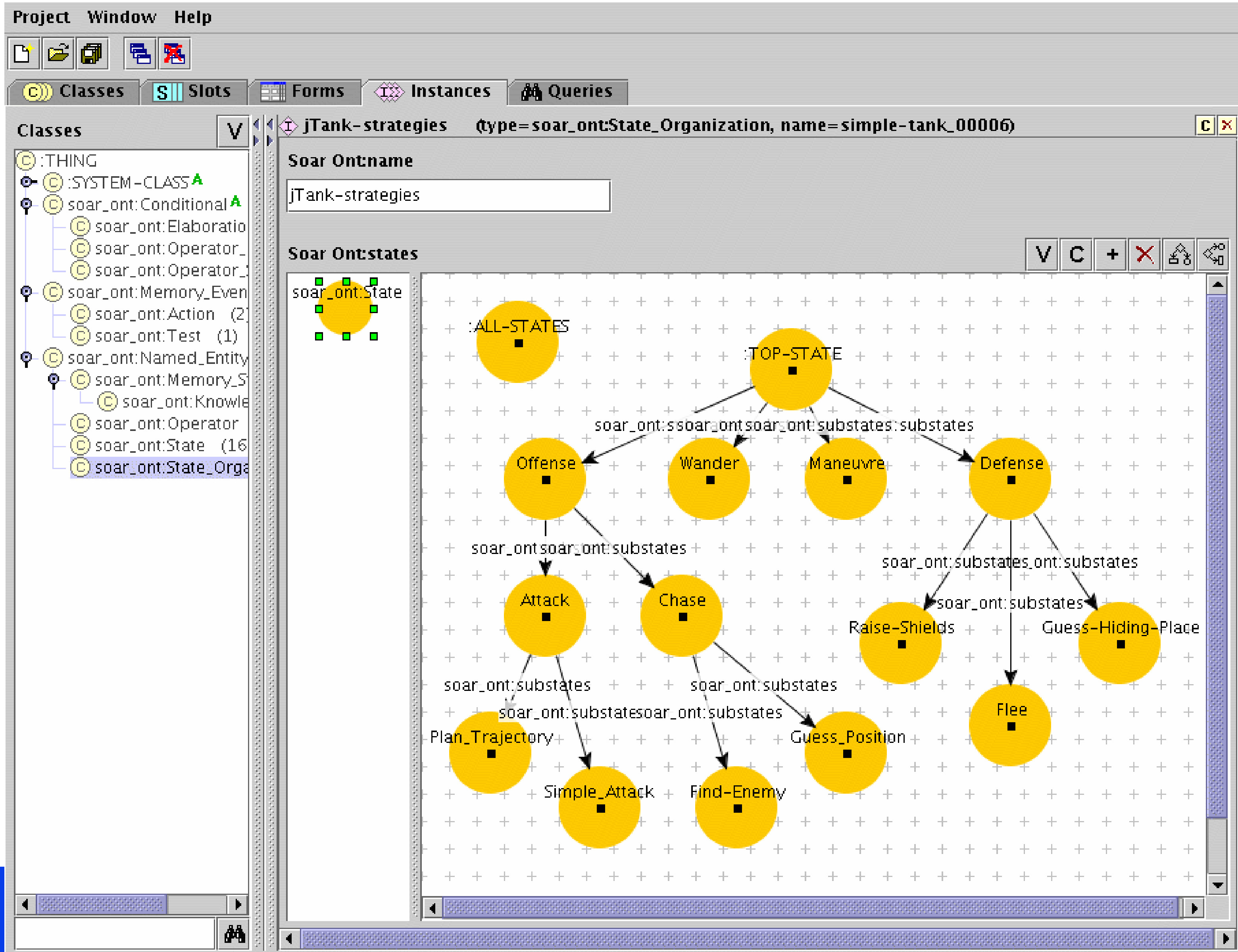
Architecture



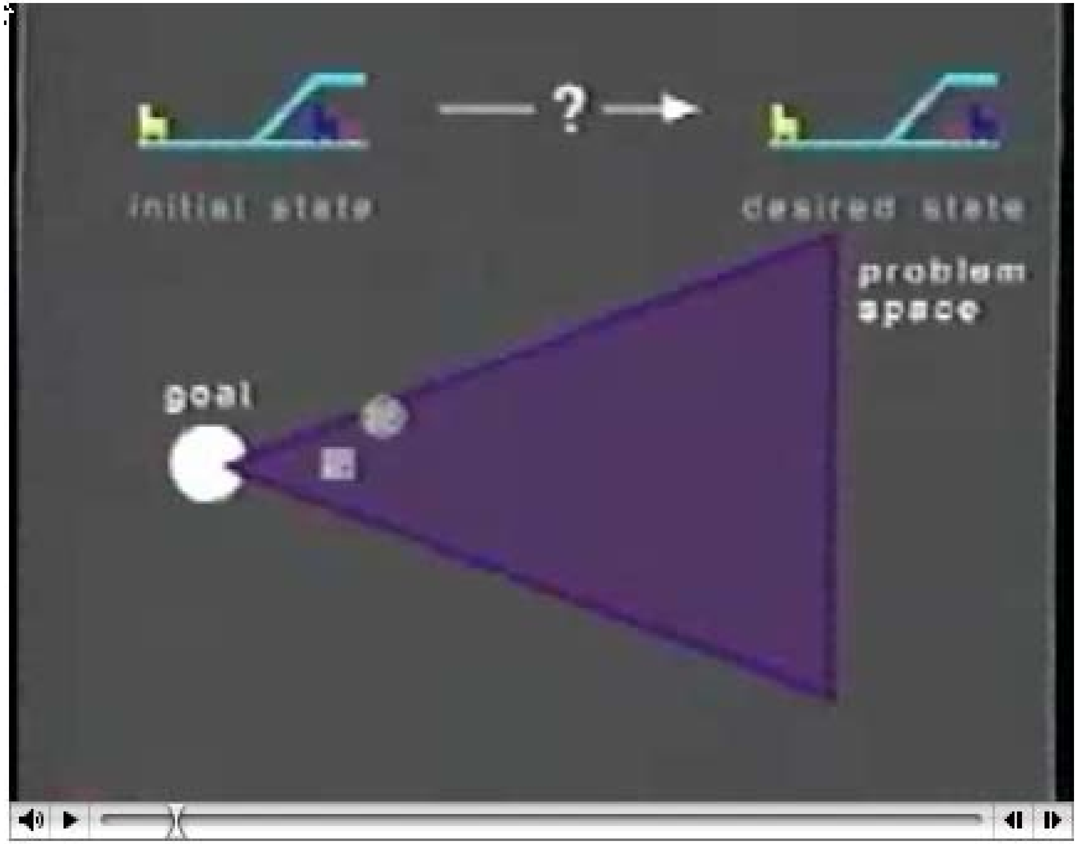
Features:

- Captures Design Documentation
- Namespacing!
- Support for Global Knowledge Bases
- Support for Importing Domain Ontologies and Model Extensions
- Horn Clauses replace Soar Syntax
- Graphical State Layout
- Exists, pre-alpha

Demo Available Thursday Night



HLBRL (3): VISTA Display Designed for Declarative Representation



This exemplar taken from the Soar video (1994), acs.ist.psu.edu/papers/soar-mov.mpg

Why Will This One Work?

- Principled design based on a theory of knowledge (PSCM, roughly and extended)
- New payoff - explanations
- Software engineering principles
  - Modularity
  - Software reuse
  - Design patterns
- No lost expressiveness, extendable by users
- User base lined up for feedback
- Designed with usability in mind

Need: Analysis of Soar Explanation Elements

- *Deconstruction of Soar architecture to identify explanatory elements*

  [Due Aug 03]

- *Design-based analysis of CGF explanation-seeking questions*

  [Haynes, Soar 23, 12 experts x 1 hours]

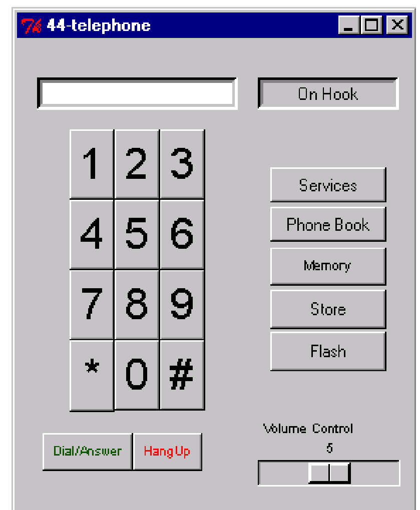
HLBRL (2): Concurrent Verbal Protocol to Explain Model

acs.ist.psu.edu/speechsynth/TextAloud.html

- *Next step: better voice, better prose, evaluation*
  [Aug 03 & repeated, Councill & Ritter]

HLBRL (6): Models Interact with Interface Directly

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



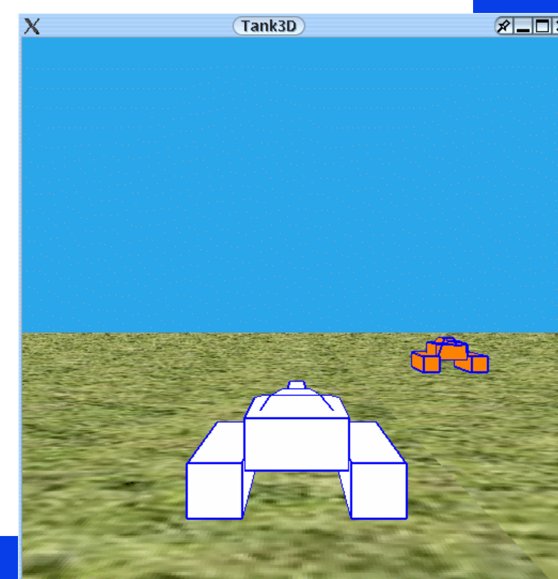
- Needs support in HLBRL compiler
- Avoids instrumenting interfaces (1/2 of code?, Myers, 1992)

HLBRL (4): jTank Microworld

- For testing explanation of dynamic, adversarial models
- Java, thus distributable across multiple machines
- Supports multiple players on multiple machines
- *First person view for human players*

QuickTime™ and a
3ivx D4 4.0.4 decompressor
are needed to see this picture.

acs.ist.psu.edu/jTank



HLBRL (5): Users to Use, Test, Expand jTank World

- IST 402: Models of human behaviour
- Microworld to understand, create, and exercise adversarial Soar models
- *Will explore usability aids, how to explain behavior, and when to interrupt users* *[Ritter & Councill & TA, Sept 03]*

More Articulate and More Understandable Soar

- High-level language supports explanation from declarative representation + rationale
  - With multi-media delivery
  - Improved developer productivity
- Microworld for exploring these issues
  - Audience of users arranged

In process

References (📖 acs.ist.psu.edu/papers/)

- Allsopp, D. J. (2002). *The specification and storage of military task knowledge to support the production of computer generated forces*. PhD thesis, RMCS, Cranfield University, UK,
- 📖 Avraamides, M., & Ritter, F. E. (2002). Using multidisciplinary expert evaluations to test and improve cognitive model interfaces. In *Proceedings of the 11th Computer Generated Forces Conference*. 553-562, 02-CGF-100. Orlando, FL: U. of Central Florida.
  One of seven papers selected by the Conference Program Committee for the Recommended Reading List from the 11th Conference on Computer-Generated Forces and Behavior Representation. www.sisostds.org/conference/View_Public_Reading.cfm?Phase_ID=2
- 📖 Councill, I. G., Haynes, S. R., & Ritter, F. E. (2003). Explaining Soar: Analysis of existing tools and user information requirements. In *Proceedings of the Fifth International Conference on Cognitive Modeling*. 63-68. Bamberg, Germany: Universitats-Verlag Bamberg.
- 📖 Kalus, T., & Ritter, F. E. (2003). The Psychological Soar Tutorial. In *Proceedings of the Fifth International Conference on Cognitive Modeling*. 318. Bamberg, Germany: Universitats-Verlag Bamberg.
- 📖 Ritter, F. E. (2003). Soar. In L. Nadel (Ed.), *Encyclopedia of cognitive science.* vol. 4, 60-65. London: Nature Publishing Group. [A006.pdf]
- 📖 Ritter, F. E., Shadbolt, N. R., Elliman, D., Young, R., Gobet, F., & Baxter, G. D. (2003). *Techniques for modeling human and organizational behaviour in synthetic environments: A supplementary review*. Wright-Patterson Air Force Base, OH: Human Systems Information Analysis Center.
- 📖 Shah, K., Rajyaguru, S., St. Amant, R., & Ritter, F. E. (2003). Connecting a cognitive model to dynamic gaming environments: Architectural and image processing issues. In *Proceedings of the Fifth International Conference on Cognitive Modeling*. 189-194. Bamberg, Germany: Universitats-Verlag Bamberg.
- Yost, G. R. (1993). Acquiring knowledge in Soar. *IEEE Expert, 8*(3), 26-34.