Adaptive Computer-Based Tutoring in an ADL Context

Dr. Scott A. Douglass
Cognitive Models and Agents Branch
Air Force Research Laboratory
Outline

• Technical Underpinnings of Tutors
  – Two types of tutors
  – Representations and processes in tutors
• Example Tutor
  – Task overview
  – How current AFRL research supports tutor development
    • Technically realizing activity monitoring
    • Net-Centric execution
    • Performance monitoring
• Long-Term Vision
Example

- Allowed actions are demonstrated
- Anticipated errors are demonstrated and associated with bug messages and hints
- Remaining errors block progress
- Minimal programming
- Limited re-usability

Cognitive

- Models support activity monitoring
- Decomposition of task into representations of knowledge and skill supports performance monitoring and the tailoring of instructional material
- Impact of repetition and inactivity on learning can be predicted and explained
- Models explain differences between instructional approaches
Technical Underpinnings of Tutors
Representations Underlying a Cognitive Tutor

• The task of interest
  – Task environment
  – Possible actions Interface Module

• The trainee
  – Current knowledge
  – Current goals Student Module
  – Recent actions
  – Knowledge/skill strengths and weaknesses

• Instruction
  – Optimal actions Expert Module
  – Pedagogically effective actions Tutor Module
  – Intervention alternatives
2. Information in a “Character Readout”

3. Interactions with interface and synthetic teammates through function-key panel

4. Information in a “Track Report Area”

5. Entry of “Type” and “Threat” identifications

1. Track Selections (visual assessments)
• **Task Selection**
  • Help trainee choose next unit task to execute

• **Task Execution**
  • Help trainee execute the current unit task
Where Cognitive Models and Agents are Going
The Challenges of Scale and Integration

Current Successes...

Modeling the Cognitive Processes Underlying Spatial Rotation
- Models Execute in the Same Application as the Cognitive Architecture

Developing Challenges...

Modeling Sense-Making as Abduction-Based Inquiry in a Cyber Effects Testbed
- Agents Integrated into a Multi-Formalism M&S Environment

- Moving models to synthetic task environments
- Modeling components of cyber/physical/human systems
  - Facing **scaling** and **integration** challenges

Making sense of and defeating software protections
Large-Scale Cognitive Modeling (LSCM) Initiative
DSL Development using Model Integrated Computing


1. Meta-Modeling -- DSL Development

2. Modeling -- DSL Use

3. Model Integration and Simulation -- DSL Transformation, Integration and Execution

\[
SYS \equiv \langle T, X, \Omega, Q, \delta, Y, \lambda \rangle
\]

- Time base
- Input set
- State
- State set
- Transition function
- Output set
- Output function

- General Systems Theory
- DEVS: Discrete Event Systems
- HLA/DEVS Integration

Wymore, A. W. (1967)
“A Mathematical Theory of Systems Engineering the Elements.”

Zeigler, Kim, & Praehofer (2000)
“Theory of Modeling and Simulation.”
An Example Cognitive Modeling DSL
Abstract/Concrete Syntaxes based in ACT-R; Semantic Anchoring in DEVS

- Formal Rigor
- Model Re-Usability
- Enhanced Interoperability
  - DoD Architectural Framework
  - DEVS-HLA
- Advanced M&S
  - Net-Centric Execution!

- Cognitive Fidelity
- Domain Concepts/Relations
- Human Modeling
  - Motor Behavior
  - Visual Behavior
  - Associative Memory
**M&S Framework of the LSCM Initiative**

**Advanced M&S in a Service-Oriented Architecture (SOA)**

- **DEVS**: discrete event system formalism
- **DEVSMML**: DEVS modeling language
- **DSL**: domain-specific language
- **M2DEVSM**: model to DEVS

**Key Components**

- **DEVS Modeling Language (DEVS PIM DSL)**
  - (Atomic and Coupled models)
- **Net-centric Infrastructure (SOA)**
- **DEVS / JAVA**: 192.168.1.100
- **DEVS / C++**: 192.168.1.101
- **DEVS / .NET**: 192.168.1.101
- **Non -DEVS**: e.g. MATLAB

**Transformations**

- **M2M transformation**
- **M2DEVSM transformation**

**Architecture**

- **End User client**
- **Server-side architecture**

**Notes**

- **M2M**: model to model
- **PIM**: platform independent model
- **SOA**: service oriented architecture
Behavior Models in Example DSL
Formal Representations that Recognize and Generate Behavior
Behavior Models in Example DSL
Formal Representations that Recognize and Generate Behavior
Behavior Models in Example DSL
Formal Representations that Recognize and Generate Behavior
Behavior Models in Example DSL
Formal Representations that Recognize and Generate Behavior
As Much Cognitive Fidelity as Necessary
“Cognitively Thin” Tutors can be Developed Quickly and then Scale

Increasingly more adherence to and use of cognitive architecture in behavior models

Example

Cognitive

- Track hook action recognized
- Another track hook recognized
- Track remains hooked
- Off target click leading to deselection

Track Hooked

- Track hook action recognized
- Another track hook recognized
- EWS request/unit-task recognized
- Help requested
- Identification/unit-task recognized
- Some other unit-task recognized

EWS Requested

- Class AMP
- Classification completed
- Air ID started
- Classification cancelled

Primary ID

- Track Manager
- Classification cancelled
- Track entry cancelled
- Track entry started
- Class AMP started
- Class AMP cancelled
- ID as "friendly"
- Help requested
- Primary ID cancelled
- Primary ID started

Air ID

Comm Non-MI classification started
M2M Transformations and Execution
Minimizing Programming and Platform Dependence

Code

DEVS

DSL
M2M Transformations and Execution
Supporting the Execution and Control of Agents
6.719  {hook_noticed, 6040}
13.766 {deliver_intervention, wav_attempt_ews}
17.078 {keypress_noticed, f10}
19.516 {show-tra-info, "BRG, 203, ARINC564, ON", 6040, requested_ews, positive}
19.516 {keypress_noticed, f1}
19.516 {report_unit_task, ews_request, 6040}
26.562 {deliver_intervention, wav_use_ews_to_id}
33.609 {deliver_intervention, wav_use_ews_to_id}
40.656 {deliver_intervention, wav_use_ews_to_id}
43.125 {keypress_noticed, f6}
...
50.016 {keypress_noticed, f1}
50.016 {report_unit_task, comm/non_mil_id, 6040}
50.016 {report_strategy, single_ews_id}
Cognitive Agents and ADL in the GIG

- **Challenges of the future**
  - Heterogeneity
    - Increasing interoperability
  - Change
    - Responding to changing requirements

- **Potential of SOAs**
  - Leverage existing systems
  - Facilitate integration
  - Manage complexity
  - Change pace/cost of development

---

DEVS Middleware
(Standards compliant API)

**DEVS / JAVA**
192.168.1.100

**DEVS / C++**
192.168.1.101

**DEVS / .NET**
192.168.1.101

**DEVS / SOA**

**Non-DEVS**
eg. MATLAB

**M2M transformation**

**End User client**

**Server-side architecture**

**M2DEVS transformation**

**M2DEVSML transformation**

---

**SOA Infrastructure**

- Service Orchestration
- Service Management
- Service Governance
- Service Security
- Service Policies
- Service Communication (Messaging)
- Information Assurance Services
- Mediation Services
- Net-centric Services
- Net-centric Infrastructure (SOA)
- Other Services
- Policy Services
- Directory Services
- Collaboration Services
- Imagery Exploitation
- Warfighting Applications
- Logistics
- Financial Management
- Human Resources Management

**Defense Intelligence Applications**

**Applications, Services, Systems**

**NetOps**

---

**DEVS Modeling Language (DEVS / PIM / DSL)**
(Atomic and Coupled models)
Future of ITS in ADL
ITS Built using DSLs and Executing in Net-Centric SOAs

- ITS development in a net-centric cognitive modeling framework
  - Task, student, expert, and tutor modules specified in DSLs
  - Transformation of DSLs to executable artifacts
  - Net-centric execution
- Authoring environment that turns an ITS author into an end-user
  - DSLs tailored to the needs of domain experts
  - Re-use of behavior models
- ITS components integrated in a SOA
  - Pedagogical agent
  - Task environment
  - Performance repository
  - Instruction management system
- Standards
  - Development
  - Interoperability
  - Delivery/deployment

*Training/ADL Applications*