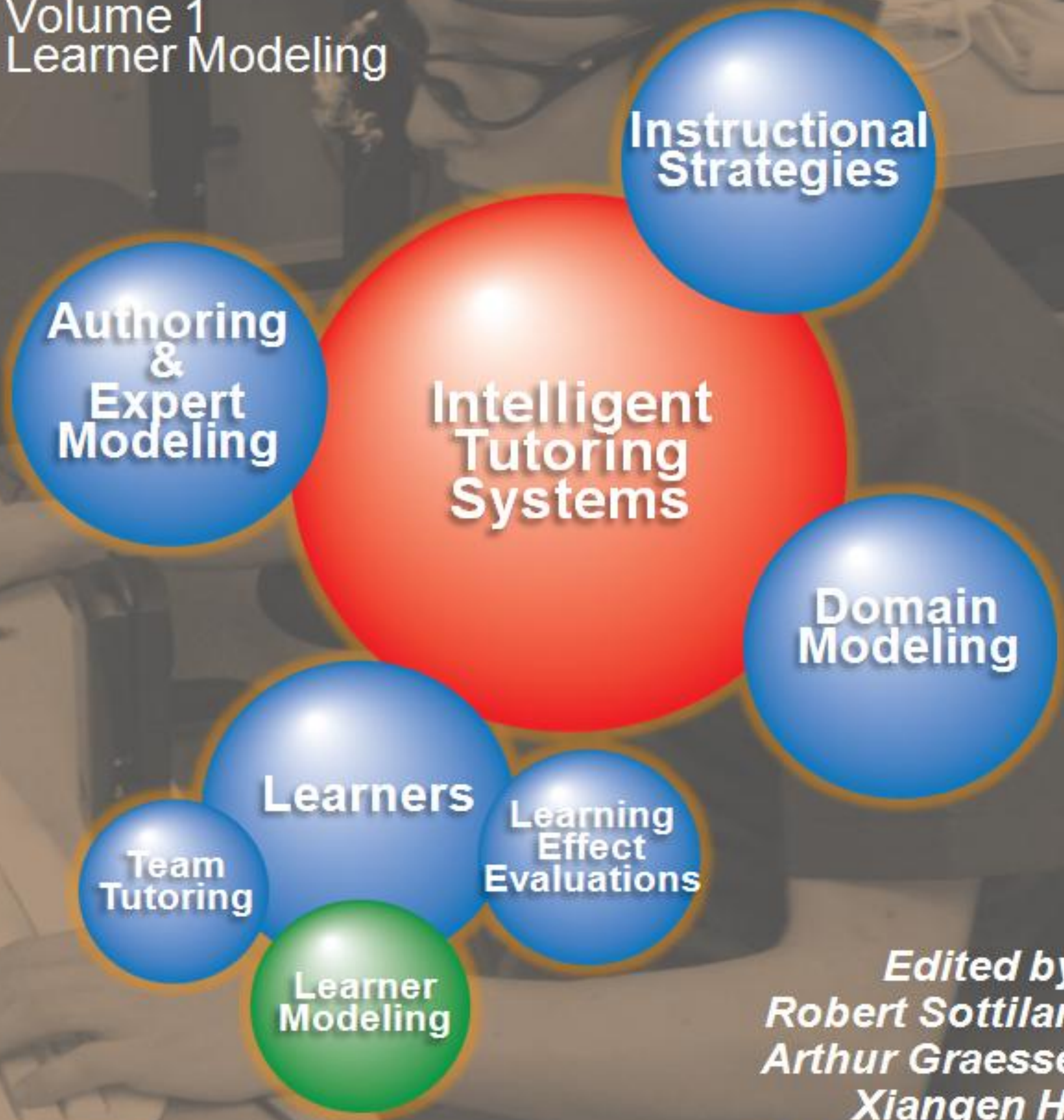
Design Recommendations for Intelligent Tutoring Systems

Volume 1
Learner Modeling


Instructional Strategies
Authoring & Expert Modeling
Intelligent Tutoring Systems
Domain Modeling
Learners
Team Tutoring
Learning Effect Evaluations
Learner Modeling

Edited by:
Robert Sottilare
Arthur Graesser
Xiangen Hu
Heather Holden

A Book in the Adaptive Tutoring Series

Design Recommendations for Intelligent Tutoring Systems

Volume 1
Learner Modeling

Edited by:
Robert Sottilare
Arthur Graesser
Xiangen Hu
Heather Holden

A Book in the Adaptive Tutoring Series

Printed in the United States of America
First Printing, July 2013
First Printing (errata addressed), August 2013

U.S. Army Research Laboratory
Human Research & Engineering Directorate
SFC Paul Ray Smith Simulation & Training Technology Center
Orlando, Florida

International Standard Book Number: 978-0-9893923-0-3

We wish to acknowledge the editing and formatting contributions of Carol Johnson, ARL

Dedicated to current and future scientists and developers of adaptive learning technologies

Contents

Preface

Robert A. Sottilare[1], Arthur C. Graesser[2], Xiangen Hu[2], and Heather Holden[1], Eds.

U.S. Army Research Laboratory - Human Research and Engineering Directorate[1]

University of Memphis Institute for Intelligent Systems[2]

This book is the first in a planned series of books that examine key topics (e.g., learner modeling, instructional strategies, authoring, domain modeling, learning effect, and team tutoring) in intelligent tutoring system (ITS) design through the lens of the Generalized Intelligent Framework for Tutoring (GIFT; Sottilare, Brawner, Goldberg, and Holden, 2012), a modular, service-oriented architecture created to develop standards for authoring, managing instruction, and analyzing the effect of ITS technologies.

This preface introduces tutoring functions, provides learner modeling examples, and examines the motivation for standards for the design, authoring, instruction, and analysis functions within ITSs. Next, we introduce GIFT design principles, and finally, we discuss how readers might use this book as a design tool. We begin by examining the concept of learner modeling.

Learner modeling (also known as student modeling or user modeling) is one of the major components of ITS. Learner modeling is a key to the computer-based tutor's understanding of the learner. Comprehensive, real-time modeling of the learner is a critical element in the design and development of truly adaptive tutoring systems that can tailor tutoring experiences to the needs of the individual learner and teams of learners.

It is generally accepted that an ITS has four major components (Elson-Cook, 1993; Nkambou, Mizoguchi & Bourdeau, 2010; Graesser, Conley & Olney, 2012; Psotka & Mutter, 2008; Sleeman & Brown, 1982; VanLehn, 2006; Woolf, 2009): The domain model, the student model, the tutoring model, and the user-interface model. GIFT similarly adopts this four-part distinction, but with slightly different corresponding labels (domain module, learner module, pedagogical module, and tutor-user interface) and the addition of the sensor module, which can be viewed as an expansion of the user interface.

(1) The domain model contains the set of skills, knowledge, and strategies of the topic being tutored. It normally contains the ideal expert knowledge and also the bugs, mal-rules, and misconceptions that students periodically exhibit.

(2) The learner model consists of the cognitive, affective, motivational, and other psychological states that evolve during the course of learning. It is often viewed as an overlay (subset) of the domain model, which changes over the course of tutoring. For example, "knowledge tracing" tracks the learner's progress from problem to problem and builds a profile of strengths and weaknesses relative to the domain model (Anderson, Corbett, Koedinger & Pelletier, 1995). An ITS may also consider psychological states outside of the domain model that need to be considered as parameters to guide tutoring.

(3) The tutor model (also known as the pedagogical model or the instructional model) takes the domain and learner models as input and selects tutoring strategies, steps, and actions on what the tutor should do next in the exchange. In mixed-initiative systems, the learners may also take actions, ask questions, or request help (Aleven, McClaren, Roll & Koedinger, 2006; Rus & Graesser, 2009), but the ITS always needs to be ready to decide "what to do next" at any point and this is determined by a tutoring model that captures the researchers' pedagogical theories.

(4) The user interface interprets the learner's contributions through various input media (speech, typing, clicking) and produces output in different media (text, diagrams, animations, agents). In addition to the conventional human-computer interface features, some recent systems have incorporated natural language interaction (Graesser et al., 2012; Johnson & Valente, 2008), speech recognition (D'Mello, Graesser & King, 2010; Litman, 2013), and the sensing of learner emotions (Baker, D'Mello, Rodrigo & Graesser, 2010; D'Mello & Graesser, 2010; Goldberg, Sottilare, Brawner, Holden, 2011).

The designers of the learner model need to decide what content, fields, variables, and parameters need to be included in the representation. The representation needs to be complete with respect to handling the distinctions made in the domain model, tutoring model, and user interface. Such representations vary in grain size, reflecting the complexity of the ITS. There is a comparatively small number of distinctions made in conventional computer-based training (O'Neil & Perez, 2003). For example, a simple system would just keep track of whether the learner has mastered (yes versus no) a set of N learning objects in the curriculum and the objects in a curriculum would be ordered theoretically, perhaps from simple to complex or along a prerequisite ladder (Gagne, 1985). The tutoring module would select the next learning object that the learner has not mastered and places that as lowest/earliest in the ordering. This kind of simple system may go a long way. However, ITSs presume to go a large step further in grain size and adaptability. Some of these ITSs are listed below, but there are many others that can be discussed at the workshop. One foundational question is whether the increased grain size and adaptability has an incremental return on investment with respect to learning gains.

Learner Modeling Examples

The following sections briefly describe four examples of learner modeling that have been developed and tested in contemporary ITSs.

Knowledge Tracing in the Cognitive Tutor

This approach to learner modeling tracks the learner's progress from problem to problem and builds a profile of strengths and weaknesses relative to the production rules (Anderson et al., 1995). A production rule is an "IF<state>THEN<action>" expression that specifies that a particular action, step, or cognitive event occurs in a particular state of the task or cognition. Information from knowledge tracing can be presented as a *skillometer*, a visual graph of the learner's success in each of the monitored skills related to solving problems in a step by step fashion. The skillometer is updated as the learner performs correct actions, commits errors, and requests a hint. Step-by-step knowledge tracing is incorporated in a number of tutors in the Pittsburgh Science of Learning Center (Aleven et al., 2006; Anderson et al., 1995; Heffernan, Koedinger & Razzaq, 2008; Ritter, Anderson, Koedinger & Corbett, 2007; VanLehn, 2006).

Constraint-based Modeling

In constraint-based tutors, a good solution is represented as a declarative structure and the learner's actions are compared with these constraints (Mitrovic, Martin & Suraweera, 2007; Ohlson, 1992). Each constraint is a declarative statement composed of a relevance condition (R) and a satisfaction condition (S). The relevance condition specifies when the constraint is relevant and only in these conditions is the state constraint meaningful. The satisfaction condition specifies whether the state constraint has been violated. A relevant, satisfied state constraint corresponds to an aspect of the correct solution. A relevant, unsatisfied state constraint indicates a flaw in the solution. Learner modeling is tracked by considering what constraints are followed as learners solve problems. Successful constraint-based tutors include Structured Query Language (SQL) tutor, *Knowledge-based Entity Relationship Modeling Intelligent Tutor* (KERMIT), and Addison-Wesley's *Database Place* (Mitrovic, Martin & Suraweera, 2007).

Knowledge Space Models

Knowledge space modeling underlies the Assessment and Learning in Knowledge Spaces (ALEKS) mathematics tutor (Doignon & Falmagne, 1999; Hu et al., 2012). The domain model of knowledge space theory is a large number of possible knowledge states on a topic, whereas the learner model is a record of which of the knowledge states are mastered, essentially a fine-grained overlay model. A learner's competence is reflected in the types of problems that the learner is capable of solving (among 250–350 problems), given the profile of knowledge states mastered among millions of possible states. Bayesian statistics are used to select the next problem to work on that is sensitive to the learner's competence by filling in deficits and correcting misconceptions. If a learner solves the next problem correctly, then each knowledge state containing that problem incrementally increases in probability; if the learner answers incorrectly, then the knowledge states are decreased in probability. Categories of skills are represented in a pie chart that reflects the competence of the learner.

Expectation and Misconception Tailored Dialogue

This type of learner modeling is typical for ITSs that help learners learn by holding a conversation in natural language, such as *AutoTutor* or *Why-Atlas* (Graesser et al., 2012; VanLehn et al., 2007). An answer to a question is a set sentence-like expectation (good answer), but the tutor also anticipates the learner articulating misconceptions (errors). An expectation or misconception is scored as being expressed by a learner if the learner articulates it in natural language with a high enough semantic match. Semantic matches can be assessed by a number of methods in computational linguistics, such as content word overlap, latent semantic analysis, regular expressions, semantic entailment, or Bayesian statistics (Cai et al., 2011; Graesser et al., 2007; Rus et al., 2009; VanLehn et al., 2007). When the total set of problems is considered, there is a universal set of expectations (called principles or facets) and misconceptions that are relevant to the various problems. These principles can be tracked over problems and guide the selection of the next problem to work on.

Motivations for Intelligent Tutoring System Standards

An emphasis on self-regulated learning has highlighted a requirement for point-of-need training in environments where human tutors are either unavailable or impractical. ITSs have been shown to be as effective as expert human tutors (VanLehn, 2011) in one-to-one tutoring in well-defined domains (e.g., mathematics or physics) and significantly better than traditional classroom training environments. ITSs have demonstrated significant promise, but fifty years of research have been unsuccessful in making ITSs ubiquitous in military training or the tool of choice in our educational system. Why?

The availability and use of ITSs have been constrained by their high development costs, their limited reuse, a lack of standards, and their inadequate adaptability to the needs of learners (Picard, 2006). Their application to military domains is further hampered by the complex and often ill-defined environments in which our military operates today. ITSs are often built as domain-specific, unique, one-of-a-kind, largely domain-dependent solutions focused on a single pedagogical strategy (e.g., model tracing or constraint-based approaches) when complex learning domains may require novel or hybrid approaches. Therefore, a modular ITS framework and standards are needed to enhance reuse, support authoring, optimize instructional strategies, and lower the cost and skillset needed for users to adopt ITS solutions for training and education. It was out of this need that the idea for GIFT arose.

GIFT has three primary functions: authoring, instructional management, and analysis. First, it is a framework for authoring new ITS components, methods, strategies, and whole tutoring systems. Second, GIFT is an instructional manager that integrates selected tutoring principals and strategies for use in ITSs. Finally, GIFT is an experimental testbed to analyze the effectiveness and impact of ITS components, tools, and methods. GIFT is based on a learner-centric approach with the goal of improving linkages in the adaptive tutoring learning effect chain (Figure P-1).

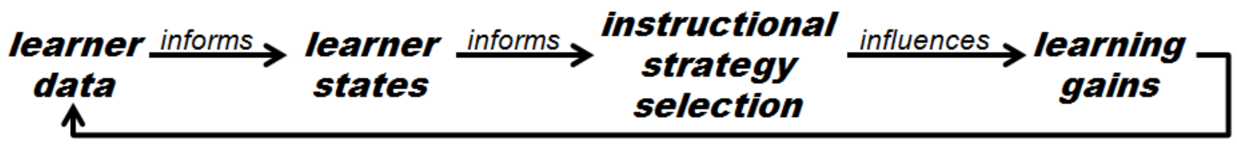


Figure P-1. Adaptive Tutoring Learning Effect Chain (Sottilare, 2012)

A deeper understanding of the learner's behaviors, traits, and preferences (learner data) collected through performance, physiological and behavioral sensors, and surveys will allow for more accurate evaluation of the learner's states (e.g., engagement level, confusion, frustration), which will result in a better and more persistent model of the learner. To enhance the adaptability of the ITS, methods are needed to accurately classify learner states (e.g., cognitive, affective, psychomotor, social) and select optimal instructional strategies given the learner's existing states. A more comprehensive learner model will allow the ITS to adapt more appropriately to address the learner's needs by changing the instructional strategy (e.g., content, flow, or feedback). An instructional strategy that is better aligned to the learner's needs is more likely to positively influence their learning gains. It is with the goal of optimized learning gains in mind that the design principles for GIFT were formulated.

GIFT Design Principles

The methodology for developing a modular, computer-based tutoring framework for training and education considered major design goals, anticipated uses, and applications. The design process also looked at enhancing one-to-one (individual) and one-to-many (collective or team) tutoring experiences beyond the state of practice for ITSs today. A significant focus of the GIFT design was on domain-dependent elements in the domain module. This was done to allow large-scale reuse of the remaining GIFT modules across different training domains and thereby reduce the development costs for ITSs.

One design principle adopted in GIFT is that each module should be capable of gathering information from other modules according to the design specification. Designing to this principle resulted in standard message sets and message transmission rules (i.e., request-driven, event-driven, or periodic transmissions). For instance, the pedagogical module is capable of receiving information from the learner module to develop courses of action for future instructional content to be displayed, manage flow and challenge level, and select appropriate feedback. Changes to the learner's state (e.g., engagement, motivation, or affect) trigger messages to the pedagogical module, which then recommends general courses of action (e.g., ask a question or prompt the learner for more information) to the domain module, which provides a domain-specific intervention (e.g., what is the next step?).

Another design principle adopted within GIFT is the separation of content from the executable code (Patil & Abraham, 2010). Data and data structures are placed within models and libraries, while software processes are programmed into interoperable modules. Efficiency and effectiveness goals (e.g., accelerated learning and enhanced retention) were considered to address the time available for military training and the renewed emphasis on self-regulated learning. An outgrowth of this emphasis on efficiency and effectiveness led Dr. Sottilare to seek external collaboration and guidance. In 2012, U.S. Army Research Laboratory (ARL) with the University of Memphis developed advisory boards of senior tutoring system scientists from academia and government to influence the GIFT design goals moving forward. An advisory board for learner modeling was completed in September 2012, and future boards are planned for instructional strategy design, authoring and expert modeling, learning effect evaluations, and domain modeling.

Design Goals and Anticipated Uses

GIFT may be used as any of the following:

1. An architectural framework with modular, interchangeable elements and defined relationships

2. A set of specifications to guide ITS development

3. A set of exemplars instantiating GIFT to support authoring and ease-of-use

4. A technical platform or testbed for guiding the development of concrete systems

These use cases have been distilled down into the three primary functional areas, or *constructs*: authoring, instructional management, and analysis. Discussed below are the purposes, associated design goals, and anticipated uses for each of the GIFT constructs.

GIFT Authoring Construct

The purpose of the GIFT authoring construct is to provide technology (tools and methods) to make it affordable and easier to build ITSs and ITS components. Toward this end, a set of extensible markup language (XML) configuration tools continues to be developed to allow for data-driven changes to the design and implementation of GIFT-generated ITSs. The design goals for the GIFT authoring construct have been adapted from Murray (1999, 2003) and Sottilare & Gilbert (2011). The GIFT authoring design goals are as follow:

- Decrease the effort (time, cost, and/or other resources) for authoring and analyzing ITSs by automating authoring processes, developing authoring tools and methods, and developing standards to promote reuse.

- Decrease the skill threshold by tailoring tools for specific disciplines (e.g., instructional designers, training developers, and trainers) to author, analyze, and employ ITS technologies.

- Provide tools to aid designers/authors/trainers/researchers in organizing their knowledge.

- Support (structure, recommend, or enforce) good design principles in pedagogy through user interfaces, and other interactions.

- Enable rapid prototyping of ITSs to allow for rapid design/evaluation cycles of prototype capabilities.

- Employ standards to support rapid integration of external training/tutoring environments (e.g., simulators, serious games, slide presentations, transmedia narratives, and other interactive multimedia).

- Develop/exploit common tools and user interfaces to adapt ITS design through data-driven means.

- Promote reuse through domain-independent modules and data structures.

- Leverage open-source solutions to reduce ITS development and sustainment costs.

- Develop interfaces/gateways to widely used commercial and academics tools (e.g., games, sensors, toolkits, virtual humans).

As a user-centric architecture, anticipated uses for GIFT authoring tools are driven largely by the anticipated users, which include learners, domain experts, instructional system designers, training and tutoring system developers, trainers and teachers, and researchers. In addition to user models and graphical user interfaces, GIFT authoring tools include domain-specific knowledge configuration tools, instructional strategy development tools, and a compiler to generate executable ITSs from GIFT components in a variety of formats (e.g., PC, Android, and IPad).

Within GIFT, domain-specific knowledge configuration tools permit authoring of new knowledge elements or reusing existing (stored) knowledge elements. Domain knowledge elements include learning objectives, media, task descriptions, task conditions, standards and measures of success, common misconceptions, feedback library, and a question library, which are informed by instructional system design principles that, in turn inform concept maps for lessons and whole courses. The task descriptions, task conditions, standards and measures of success, and common misconceptions may be informed by an expert or ideal learner model derived through a task analysis of the behaviors of a highly skilled user. ARL is investigating techniques to automate this expert model development process to reduce the time and cost of developing ITSs. In addition to feedback and questions, supplementary tools are anticipated to author explanations, summaries, examples, analogies, hints, and prompts in support of GIFT's instructional management construct.

GIFT Instructional Management Construct

The purpose of the GIFT instructional management construct is to integrate pedagogical best practices in GIFT-generated ITSs. The modularity of GIFT will also allow GIFT users to extract pedagogical models for use in tutoring/training systems that are not GIFT-generated. GIFT users may also integrate pedagogical models, instructional strategies, or instructional tactics from other tutoring systems into GIFT. The design goals for the GIFT instructional management construct are the following:

- Support ITS instruction for individuals and small teams in local and geographically distributed training environments (e.g., mobile training), and in both well-defined and ill-defined learning domains.

- Provide for comprehensive learner models that incorporate learner states, traits, demographics, and historical data (e.g., performance) to inform ITS decisions to adapt training/tutoring.

- Support low-cost, unobtrusive (passive) methods to sense learner behaviors and physiological measures and use these data along with instructional context to inform models to classify (in near real time) the learner's states (e.g., cognitive and affective).

- Support both macro-adaptive strategies (adaptation based on pre-training learner traits) and micro-adaptive instructional strategies and tactics (adaptation based learner states and state changes during training).

- Support the consideration of individual differences where they have empirically been documented to be significant influencers of learning outcomes (e.g., knowledge or skill acquisition, retention, and performance).

- Support adaptation (e.g., pace, flow, and challenge level) of the instruction based the domain and learning class (e.g., cognitive learning, affective learning, psychomotor learning, social learning).

- Model appropriate instructional strategies and tactics of expert human tutors to develop a comprehensive pedagogical model.

To support the development of optimized instructional strategies and tactics, GIFT is heavily grounded in learning theory, tutoring theory, and motivational theory. Learning theory applied in GIFT includes cognitive learning (Anderson & Krathwohl, 2001), affective learning (Krathwohl, Bloom, and Masia, 1964; Goleman, 1995), psychomotor learning (Simpson, 1972), and social learning (Sottilare, Holden, Brawner, and Goldberg, 2011; Soller, 2001). Aligning with our goal to model expert human tutors, GIFT considers the INSPIRE model of tutoring success (Lepper, Drake, and O'Donnell-Johnson, 1997) and the tutoring process defined by Person, Kreuz, Zwaan, and Graesser (1995) in the development of GIFT instructional strategies and tactics.

INSPIRE is an acronym that highlights the seven critical characteristics of successful tutors: Intelligent, Nurturant, Socratic, Progressive, Indirect, Reflective, and Encouraging. Graesser & Person's (1994) tutoring process includes a tutor-learner interchange where the tutor asks a question, the learner answers the question, the tutor gives feedback on the answer, then the tutor and learner collaboratively improve the quality of (or embellish) the answer. Finally, the tutor evaluates learner's understanding of the answer.

As a learner-centric architecture, anticipated uses for GIFT instructional management capabilities include both automated instruction and blended instruction, where human tutors/teachers/trainers use GIFT to support their curriculum objectives. If its design goals are realized, it is anticipated that GIFT will be widely used beyond military training contexts as GIFT users expand the number and type of learning domains and resulting ITS generated using GIFT.

GIFT Analysis Construct

The purpose of the GIFT analysis construct is to allow ITS researchers to experimentally assess and evaluate ITS technologies (ITS components, tools, and methods). The design goals for the GIFT analysis construct are the following:

- Support the conduct of formative assessments to improve learning

- Support summative evaluations to gauge the effect of technologies on learning

- Support assessment of ITS processes to understand how learning is progressing throughout the tutoring process

- Support evaluation of resulting learning versus stated learning objectives

- Provide diagnostics to identify areas for improvement within ITS processes

- Support the ability to comparatively evaluate ITS technologies against traditional tutoring or classroom teaching methods

- Develop a testbed methodology to support assessments and evaluations (Figure P-2)

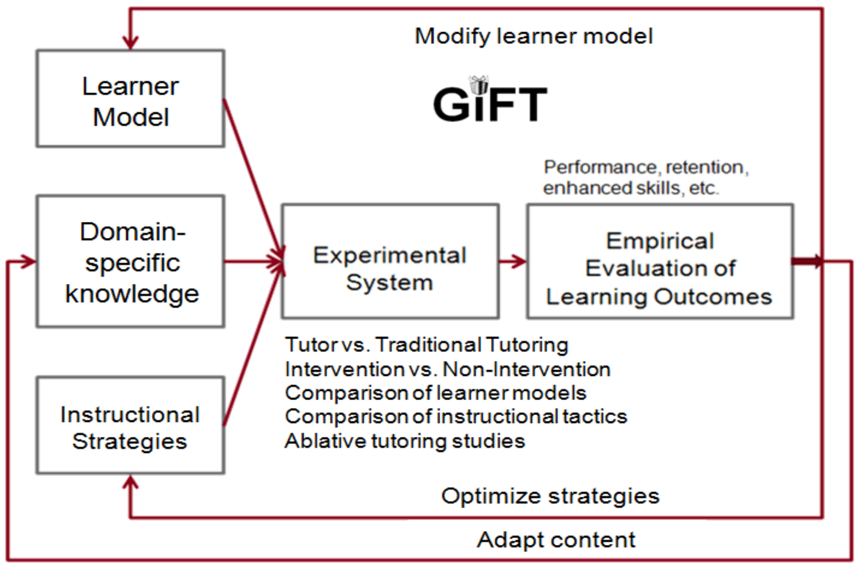

Modify learner model
GIFT
Learner Model
Domain-specific knowledge
Instructional Strategies
Experimental System
Performance, retention, enhanced skills, etc.
Empirical Evaluation of Learning Outcomes
Tutor vs. Traditional Tutoring
Intervention vs. Non-Intervention
Comparison of learner models
Comparison of instructional tactics
Ablative tutoring studies
Optimize strategies
Adapt content

Figure P-2. GIFT Analysis Testbed Methodology

Figure P-2 illustrates an analysis testbed methodology being implemented in GIFT. This methodology was derived from Hanks, Pollack, and Cohen (1993) to allow manipulation of the learner model, instructional strategies, and domain-specific knowledge within GIFT, and support analysis of artificially-intelligent agents that influence the adaptive tutoring learning effect chain. In developing their testbed methodology, Hanks et al. reviewed four testbed implementations (Tileworld, the Michigan Intelligent Coordination Experiment [MICE], the Phoenix testbed, and Truckworld) for evaluating the performance of artificially intelligent agents. Although agents have changed substantially in complexity during the past 20–25 years, the methods to evaluate their performance have remained markedly similar.

The authors designed the GIFT analysis testbed based upon Cohen's assertion (Hanks et al., 1993) that testbeds have three critical roles related to the three phases of research. During the exploratory phase, agent behaviors need to be observed and classified in broad categories. This can be performed in an experimental environment. During the confirmatory phase, the testbed is needed to allow more strict characterizations of agent behavior to test specific hypotheses and compare methodologies. Finally, in order to generalize results, measurement and replication of conditions must be possible. Similarly, the GIFT analysis methodology (Figure P-2) enables the comparison/contrast of ITS elements and assessment of their effect on learning outcomes (e.g., knowledge acquisition, skill acquisition, and retention).

How to Use This Book

This book is organized into four sections:

I. **Fundamentals of Learner Modeling**
II. **Current Learner Modeling Tools and Methods**
III. **Emerging Learner Modeling Concepts**
IV. **Future Learner Modeling Concepts**

The *Fundamentals of Learner Modeling* section provides an overview of learner modeling terms and concepts along with discussion topics, and a review of the learner modeling literature. The *Current Learner Modeling Tools and Methods* section reviews current learner modeling tools and methods and provides design recommendations for GIFT and adaptive ITSs. The *Emerging Learner Modeling Concepts* section analyzes emerging learner modeling concepts and discusses their potential impact on design recommendations for GIFT and adaptive ITS. Finally, the *Future Learner Modeling Concepts* section projects how ITSs might be applied in the future and provides design recommendations to realize innovative capabilities in GIFT and adaptive ITSs.

Chapter authors in each section were carefully selected for participation in this project based on their expertise in the field as ITS scientists, developers, and practitioners. *Design Recommendations for Intelligent Tutoring Systems: Learner Modeling (Volume I)* is intended to be a design resource as well as community research resource that can be of significant benefit as the following:

- ***An educational resource for developing ITS scientists:*** Section I provides a wealth of information about ITS concepts and design, and presents an in-depth review of the learner modeling literature.

- ***A roadmap for ITS research opportunities:*** Sections II, III, and IV present current, emerging, and future concepts about learner modeling. This sampling of authors' perspectives is based on hundreds of cumulative years of experience in the ITS research domain and identifies significant gaps in current and emerging ITS technology (tools and methods). Each of these gaps points to yet unanswered research questions.

- ***A roadmap to the development and application of GIFT****:* As noted previously, GIFT is an open-source, publically available ITS architecture that is intended to make it easy to author ITSs; reduce the cost of ITS development by promoting reuse; automatically manage instruction based on best pedagogical practices; and allow scientists to compare and contrast evolving ITS capabilities to determine future best practices. As this book outlines issues and challenges associated with learner modeling, it also provides guidelines on how GIFT might be designed to address identified capability gaps. Future volumes of the *"Design Recommendations for Intelligent Tutoring Systems"* book series will provide insight to other ITS design domains including instructional strategy and tactics design, authoring and expert modeling, domain modeling, learning effect assessment, and team tutoring design. We encourage readers to become members of the GIFT community to build on its existing capabilities and support its future capabilities with us. More information on GIFT can be found by registering at www.GIFTtutoring.org. Registration provides access to GIFT source code, documentation, and related publications.

References

Aleven, V., McLaren, B., Roll, I. & Koedinger, K. (2006). Toward meta-cognitive tutoring: A model of help seeking with a cognitive tutor. *International Journal of Artificial Intelligence in Education, 16*, 101-128.

Anderson, J. R., Corbett, A. T., Koedinger, K. R. & Pelletier, R. (1995). Cognitive tutors: Lessons learned. *Journal of the Learning Sciences, 4*, 167-207.

Anderson, L. W. & Krathwohl, D. R. (Eds.). (2001). *A taxonomy for learning, teaching and assessing: A revision of Bloom's Taxonomy of Educational Objectives: Complete edition*. New York : Longman.

Baker, R.S., D'Mello, S.K., Rodrigo, M.T. & Graesser, A.C. (2010). Better to be frustrated than bored: The incidence, persistence, and impact of learners' cognitive-affective states during interactions with three different computer-based learning environments. *International Journal of Human-Computer Studies, 68*, 223-241.

Cai, Z., Graesser, A.C., Forsyth, C., Burkett, C., Millis, K., Wallace, P., Halpern, D. & Butler, H. (2011). Trialog in ARIES: User input assessment in an intelligent tutoring system. In W. Chen & S. Li (Eds.*), Proceedings of the 3rd IEEE International Conference on Intelligent Computing and Intelligent Systems* (pp.429-433). Guangzhou: IEEE Press.

D'Mello, S. & Graesser, A.C. (2010). Multimodal semi-automated affect detection from conversational cues, gross body language, and facial features. *User Modeling and User-adapted Interaction, 20*, 147-187.

D'Mello, S. K., Graesser, A. C. & King, B. (2010). Toward spoken human-computer tutorial dialogues. *Human Computer Interaction, 25*, 289-323.

Doignon, J.P. & Falmagne, J. C. (1999). *Knowledge Spaces*. Berlin, Germany: Springer.

Elson-Cook, M. (1993). Student modeling in intelligent tutoring systems. *Artificial Intelligence Review, 7*, 227-240.

Gagne, R. M. (1985). *The conditions of learning and theory of instruction* (4th ed.). New York: Holt, Rinehart & Winston.

Goldberg, B.S., Sottilare, R.A., Brawner, K.W. & Holden, H.K. (2011). Predicting Learner Engagement during Well-Defined and Ill-Defined Computer-Based Intercultural Interactions. In S. D'Mello, A. Graesser, , B. Schuller & J.-C. Martin (Eds.), *Proceedings of the 4th International Conference on Affective Computing and Intelligent Interaction (ACII 2011) (Part 1: LNCS 6974)* (pp. 538-547). Berlin Heidelberg: Springer.

Graesser, A.C., Conley, M. & Olney, A. (2012). Intelligent tutoring systems. In K.R. Harris, S. Graham & T. Urdan (Eds.), *APA Educational Psychology Handbook: Vol. 3. Applications to Learning and Teaching* (pp. 451-473). Washington, DC: American Psychological Association.

Graesser, A. C., D'Mello, S. K., Hu. X., Cai, Z., Olney, A. & Morgan, B. (2012). AutoTutor. In P. McCarthy & C. Boonthum-Denecke (Eds.), *Applied natural language processing: Identification, investigation, and resolution* (pp. 169-187). Hershey, PA: IGI Global.

Graesser, A. C., Penumatsa, P., Ventura, M., Cai, Z. & Hu, X. (2007). Using LSA in AutoTutor: Learning through mixed initiative dialogue in natural language. In T. Landauer, D. McNamara, S. Dennis & W. Kintsch (Eds.), *Handbook of latent semantic analysis* (pp. 243–262). Mahwah, NJ: Erlbaum.

Graesser, A. C. & Person, N. K. (1994). Question asking during tutoring. *American Educational Research Journal*, *31*, 104–137.

Hanks, S., Pollack, M.E. & Cohen, P.R. (1993). Benchmarks, Test Beds, Controlled Experimentation, and the Design of Agent Architectures. *AI Magazine, 14* (4), 17-42.

Heffernan,N.T., Koedinger, K.R. & Razzaq, L.(2008) Expanding the model-tracing architecture: A 3rd generation intelligent tutor for Algebra symbolization. *The International Journal of Artificial Intelligence in Education, 18*(2). 153-178.

Hu, X., Craig, S. D., Bargagliotti A. E., Graesser, A. C., Okwumabua, T., Anderson, C., Cheney, K. R. & Sterbinsky, A. (2012). The effects of a traditional and technology-based after-school program on 6th grade students' mathematics skills. *Journal of Computers in Mathematics and Science Teaching, 31*, 17-38.

Johnson, L. W. & Valente, A. (2008). Tactical language and culture training systems: Using artificial intelligence to teach foreign languages and cultures. In M. Goker & K. Haigh (Eds.), Proceedings of the Twentieth Conference on Innovative Applications of Artificial Intelligence (pp. 1632-1639). Menlo Park, CA: AAAI Press.

Krathwohl, D.R., Bloom, B.S. & Masia, B.B. (1964). *Taxonomy of Educational Objectives: Handbook II: Affective Domain*. New York: David McKay Co.

Lepper, M. R., Drake, M. & O'Donnell-Johnson, T. M. (1997). Scaffolding techniques of expert human tutors. In K. Hogan & M. Pressley (Eds), *Scaffolding learner learning: Instructional approaches and issues* (pp. 108-144). New York: Brookline Books.

Litman, D. (2013). Speech and language processing for adaptive training. In P. Durlach & A. Lesgold (Eds.), *Adaptive technologies for training and education*. Cambridge, MA: Cambridge University Press.

Mitrovic, A., Martin, B. & Suraweera, P. (2007). Intelligent tutors for all: The constraint-based approach. *IEEE Intelligent Systems, 22*(4), 38-45.

Murray, T. (1999). Authoring intelligent tutoring systems: An analysis of the state of the art. *International Journal of Artificial Intelligence in Education*, *10*(1), 98–129.

Murray, T. (2003). An Overview of Intelligent Tutoring System Authoring Tools: Updated analysis of the state of the art. In Murray, T.; Blessing, S.; Ainsworth, S. (Eds.), *Authoring tools for advanced technology learning environments* (pp. 491-545). Berlin: Springer..

Nkambou, R., Mizoguchi, R. & Bourdeau, J. (2010). *Advances in intelligent tutoring systems*. Heidelberg: Springer.

Ohlsson, S. (1992). Constraint-based student modeling. *Journal of Artificial Intelligence in Education 3*(4), 429-447.

O'Neil, H. F. & Perez, R. (Eds.). (2003). *Technology applications in education: A learning view*. Hillsdale, NJ.: Erlbaum.

Patil, A. S. & Abraham, A. (2010). Intelligent and Interactive Web-Based Tutoring System in Engineering Education: Reviews, Perspectives and Development. In F. Xhafa, S. Caballe, A. Abraham, T. Daradoumis & A. Juan Perez (Eds.), *Computational Intelligence for Technology Enhanced Learning. Studies in Computational Intelligence* (Vol 273, pp. 79-97). Berlin: Springer-Verlag.

Person, N. K., Kreuz, R. J., Zwaan, R. A. & Graesser, A. C. (1995). Pragmatics and pedagogy: Conversational rules and politeness strategies may inhibit effective tutoring. *Cognition and Instruction, 13*(2), 161–188.

Picard, R. (2006). Building an Affective Learning Companion. Keynote address at the 8th International Conference on Intelligent Tutoring Systems, Jhongli, Taiwan. Retrieved from http://www.its2006.org/ITS_keynote/ITS2006_01.pdf

Psotka, J. & Mutter, S.A. (1988). *Intelligent Tutoring Systems: Lessons Learned*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Ritter, S., Anderson, J. R., Koedinger, K. R. & Corbett, A. (2007) Cognitive Tutor: Applied research in mathematics education. *Psychonomic Bulletin & Review, 14*, 249-255.

Rus, V. & Graesser, A.C. (Eds.) (2009). The Question Generation Shared Task and Evaluation Challenge. Retrieved from http://www.questiongeneration.org/.

Rus, V., McCarthy, P.M., McNamara, D.S. & Graesser, A.C. (2009). Identification of sentence-to-sentence relations using a text entailer. *Research on Language and Computation, 7*, 209-229.

Simpson, E. (1972). The classification of educational objectives in the psychomotor domain: *The psychomotor domain*. Vol. 3. Washington, DC: Gryphon House.

Sleeman D. & J. S. Brown (Eds.) (1982). *Intelligent Tutoring Systems*. Orlando, Florida: Academic Press, Inc.

Soller, A. (2001). Supporting social interaction in an intelligent collaborative learning system. *International Journal of Artificial Intelligence in Education, 12*(1), 40-62.

Sottilare, R. & Gilbert, S. (2011). Considerations for tutoring, cognitive modeling, authoring and interaction design in serious games. *Authoring Simulation and Game-based Intelligent Tutoring workshop at the Artificial Intelligence in Education Conference (AIED) 2011*, Auckland, New Zealand, June 2011.

Sottilare, R., Holden, H., Brawner, K. & Goldberg, B. (2011). Challenges and Emerging Concepts in the Development of Adaptive, Computer-based Tutoring Systems for Team Training. *Interservice/Industry Training Systems & Education Conference*, Orlando, Florida, December 2011.

Sottilare, R.A., Brawner, K.W., Goldberg, B.S. & Holden, H.K. (2012). *The Generalized Intelligent Framework for Tutoring (GIFT)*. Orlando, FL: U.S. Army Research Laboratory – Human Research & Engineering Directorate (ARL-HRED).

Sottilare, R. (2012). Considerations in the development of an ontology for a Generalized Intelligent Framework for Tutoring. *International Defense & Homeland Security Simulation Workshop* in Proceedings of the I3M Conference. Vienna, Austria, September 2012.

VanLehn, K. (2006) The behavior of tutoring systems. *International Journal of Artificial Intelligence in Education. 16*(3), 227-265.

VanLehn, K., Graesser, A. C., Jackson, G. T., Jordan, P., Olney, A. & Rose, C. P. (2007). When are tutorial dialogues more effective than reading? *Cognitive Science*, *31*, 3-62.

VanLehn, K. (2011). The relative effectiveness of human tutoring, intelligent tutoring systems and other tutoring systems. *Educational Psychologist*, *46*(4), 197-221.

Woolf, B.P. (2009). *Building intelligent interactive tutors*. Burlington, MA: Morgan Kaufmann Publishers.

SECTION I

FUNDAMENTALS OF LEARNER MODELING

A. Graesser, Ed.

CHAPTER 1 – A Guide to Understanding Learner Models

Arthur Graesser
University of Memphis

Introduction

The core notion of a learner model is not complex intuitively. We simply need to record, represent, and track characteristics of the learner before, during, and after learning. The mission is accomplished when the learner model accommodates all of the variables that are ever considered in the history and future of intelligent tutoring systems, with all variables being adequately represented for all systems. The *theoretical* problem is that the set of variables and their representation is not a closed system, but rather grows over time as a moving target. The *practical* problem is that it is expensive to identify, track, store, update, and later retrieve the ever-growing universal set of variables. The *mapping* problem is that the alignment between the theoretical variables and computer code is often vague, incomplete, or incompatible. The *computational* problem is that complex interactions among learner variables create a combinatorial explosion that is temporally insurmountable. In light of these multiple problems, there is no alternative than to pursue sensible compromises.

The chapters in this section have identified the major challenges in developing a learner model for GIFT. Simply put, there is no consensus in how the community of researchers is to handle the theoretical, practical, mapping, and computational problems. They all offer hints toward a solution, which was their charge. It is difficult to decide whether any of them have answered the challenge. Perhaps yes. Perhaps no.

The chapter by Robson and Barr expresses the need to establish standards. They argue that the standards should be pitched at the macro level rather than the micro level. That is, the community of researchers should agree on the ingredients of major content objects (i.e., knowledge, skills, procedures) but leave it to the individual learning environments to realize the dynamics of mastering these learning objects. This would require an agreement among curriculum experts and system developers in converging on an ideal grain size that differentiates macro and micro.

The chapter by Olney and Cade recommends that researchers take stock of the pedagogical strategies offered by researchers and to identify the learning model variables that support such strategies. The Army Research Lab conducted a systematic study to identify the strategies from hundreds of studies. Olney and Cade identified the classes of learner model variables that would support these strategies. This is a sensible approach to identifying a complete set of learner variables, including cognitive, social, emotion, and motivation dimensions.

The chapter by Lesgold and Graesser raised the persistent problem of transfer. It is comparatively easy to develop a system that efficiently trains learners on the knowledge and skills of a specific learning object, but it is difficult to do so in a way that transfers to a new learning object with related knowledge and skills. Specific is easy, but general is difficult. The authors emphasize that it is absolutely critical to acquire the materials in a general way during training that transfers to a broad range of situations later on. If not, the learning episode is destined to reside in a very narrow corner of the space to be mastered.

The chapter by Pavlik, Brawner, Olney, and Mitrovic provides a serious comprehensive review of the learning models in ITS applications over the decades. They take stock of the learner models in a variety of ITS frameworks, including step-based cognitive tutors, constraint based tutors, knowledge space models, dialogue systems, and trait-based assessments. They point out the value of systems with branching architectures and identify the alternative grain sizes of the branching, as well as the consequences. They also urge the GIFT to build on the major progress that has been made on previous ITS with general architectures and empirical validation.

The four chapters in this section together have suggested a number of factors that need to be considered as GIFT takes on the challenge of learner modeling. The goal of this section introduction is to provide a landscape of relevant dimensions rather than to offer concrete solutions.

Landscape of Variables for Learner Models

The number of learner variables is potentially large and unlimited but a small number of variables is typically tracked in systems that scale up. For example, the variables tracked in educational systems throughout the country rarely go beyond attendance and 1-3 high stakes tests per year. The collection of fine grained measures and learning portfolios are boutique enterprises that interest researchers and teachers of the future. Consequently, any convergence on an intermediate, manageable set of learner variables will need to satisfy numerous political and practical constraints.

The number of learner variables being considered in ITS research has substantially increased over the decades. There are many measures of learning gains that assess changes between pretests and posttests. The efficiency of learning is measured in economic models that consider how much learning occurs per unit time. The mastery of specific knowledge, skills, and strategies is assessed at ever increasing grain sizes. The learners' engagement and persistence can be tracked by recording time on task, reactions to computer requests, non-invasive measures (e.g., eye movements, body posture), and physiological measures. The emotions and motivation of the learner are tracked by algorithms that mine the log files of the person-system interactions. These behavioral measures are arguably more valid than self-report data, such as rating scales that are contaminated by the learners' metacognitive folklore. Measures of personality, leadership, and social responsiveness are also tracked by algorithms that have evolved from the educational data mining community. Contemporary ITS applications routinely collect thousands of measures during a time span of 2 to 20 hours. The grain size of ITS measurement is currently 3 orders of magnitude beyond the data collected in school systems throughout the country.

Farming and Mining the Landscape of Variables

Researchers need to be selective when analyzing the rich log file data that tracks the learner models. The most straightforward approach is to focus on those raw or composite measures that are anticipated theoretically. This top-down approach is the perfect place to start and impresses most reviewers of academic journals and funding agencies. Large data sets can be "farmed" by researchers in a manner that systematically tests and revises theories in the face of empirical data. Data are selected and organized to test major learning theories of the day, well established ITS applications (see Pavlik et al. chapter), educational standards (such as the Common Core or certification on specialty topics), the learning strategies documented by the ARL (see Olney and Cade chapter), an existing repository of learning objects that are shared by the community (see Robson and Barr chapter), and transfer between learning objects, tasks, and subject matters (see Lesgold and Graesser chapter).

Theories are unfortunately limited and frequently not confirmed. Consequently, there is a need for bottom-up methods to discover new learner measures and patterns from the log files. During the last decade, the field has experienced the evolution of the data mining revolution. New categories of learners are revealed by clustering analyses on learners and on tasks, as well as the tracking of individual learner data over time. Longitudinal research designs (which track individuals over a long period of time) are preferred over cross sectional research designs. Sequences of events in the log files are diagnostic of specific psychological attributes. Once these patterns are discovered, they can be tracked automatically and tested further. This approach is expected to lead to the development of more sophisticated learning theories that have a better chance of scaling up.

Representing Learner Models

ITS developers are known to disagree over the representation of knowledge, skills, and strategies in the learner model. This is because the computational models are very different in the cognitive tutors, constraint-based tutors, knowledge space tutors, dialogue-based tutors, scenario-based tutors, and other classes of ITS architectures. The representation of production rules also may differ in ITS applications that adopt production rules. The primitive elements, symbolic expressions, and quantitative parameters differ among production rules systems because the subject matter applications have very different constraints.

It is too early in the history of ITS development to force researchers to adopt a particular representation of knowledge, skills, and strategies. The SCORM initiative never was able to achieve such a lofty objective, even though there were some discussions to try. Researchers are deeply wedded to their pet computational architectures and algorithms as they pursue cycle after cycle of model testing. With this context in mind, GIFT developers may consider focusing on specific exemplar ITS applications that have proven the test of time and empirical validation (see Pavlik et al. chapter). There are a limited set of successful ITS applications so this approach would be a practical first step that is within reach. New ITS applications can build upon these prototypical exemplar ITS applications. The learner models of these exemplars could be expressed in a general formal notation in addition to the actual software residing in a repository of concrete applications. The specific ITS developers would of course need to agree to annotate and release the software. Once the collection of ITS exemplars is available, the time would be ripe to identify a first-cut landscape of variables for learner modeling.

The complexity and variations in representations have left us with some significant barriers in scaling up ITS. There are not enough trained personnel to build new applications on new subject matters because of the idiosyncratic features of the ITS representations. An ideal author would have expertise in the subject matter, cognitive science, information sciences, education, ITS pedagogy, human computer interaction, and sometimes computational linguistics. Authoring tools are often created to minimize this barrier, but there have never been sufficient efforts to build high quality authoring tools. There needs to be systematic R&D on authoring tool development that is tested on personnel outside of the camp of the original ITS developers. We need an applied empirical science of authoring tool development that has analogues to research on writing or to design. To what extent are the learner model representations developed with sufficient fidelity, scope, grain-size, and level of abstraction? How much training is needed for new personnel to develop learner models for new applications? What is the time course and costs of developing new learner models?

GIFT in the Short-Term Horizon

As mentioned earlier, the contributors to this section of the book offered different recommendations for developing the learner model component of GIFT. The recommendations addressed challenges and pressure points that need attention in the roadmap ahead. This final section enumerates some actions that might be considered in the short-term horizon.

(1) A prototype has been developed that implements characteristics of GIFT, including the learner model. A systematic analysis could be conducted on the learner model variables in order to assess the extent to which they cover the variables present in the learner models of different classes of mainstream ITS applications (e.g., cognitive tutors, constraint-based tutors, knowledge space tutors, dialogue-based tutors)

and the pedagogical strategies identified by the ADL. A repository of variables could be assembled with definitions and links to alternative ITS applications.

(2) It is widely acknowledged that it is difficult to achieve successful transfer from one domain or subject matter to that of another. The field would benefit from an analysis of training strategies, representation specification, degree of abstraction, grain size, and other characteristics of learner models of ITS applications that have achieved good transfer.

(3) Shareable learning objects (such as SCORM) have been a persistent dream of many communities that want to scale up advanced learning environments. The field would benefit from a review of the successes and failures of these attempts. This includes an analysis of the level of abstraction and types of representations that are likely to be shareable and serve as standards.

(4) Authoring tools have nearly always been difficult to use for ITS as well as other learning architectures. The field could benefit from a review of empirical research that has systematically analyzed how new personnel use such tools as well as the quantity and quality of their products from the standpoint of learner modeling in particular. New empirical studies are needed that are more systematic than anecdotal.

(5) The time and costs of developing an ITS on a new subject matter has frequently been a focus of questions with respect to scaling up the ITS enterprise. The field would benefit from an economic analysis that helps answer these questions. It is important to segregate the initial up-front costs in developing initial ITS applications, incremental costs in developing new ITS applications that piggyback on existing systems, and scale-up costs after an existing system is ready to be used by thousands or millions of students.

(6) Classes of ITS are ideally tailored to different types of learning, such as strategically guided perception, memory for facts, execution of procedures, explanations of events within complex systems, principle-based prediction/forecasting, and removal of chronic misconceptions in mental models. The field would benefit from a typology and possibly a consensus on what learning environments are appropriate for each type of learning.

CHAPTER 2 – Lowering the Barrier to Adoption of Intelligent Tutoring Systems through Standardization

Robby Robson[1] and Avron Barr[2]
[1]Eduworks Corporation; [2]Aldo Ventures

Introduction

To have an impact, a learning system must be effective and must be used. In the case of ITSs, studies have repeatedly and consistently shown significant learning gains (Dodds & Fletcher, 2004; Durlach & Ray, 2011; Kulik & Kulik, 1991; VanLehn, 2011). Despite their demonstrated value and over thirty-year history (Barr, Beard & Atkinson, 1975; Sleeman & Brown, 1982), the use of ITSs remains restricted to research projects and a few commercial applications. There are success stories, but in general, such systems are not being adopted at the rates that their effectiveness would justify. As stated by Blessing, Gilbert, Ourada & Ritter (2009) in a paper on authoring model-tracing cognitive tutors, "*ITSs, including model-tracing tutors, have not been widely adopted in educational or other settings, such as corporate training. Perhaps the most successful deployment of model-tracing tutors is Carnegie Learning's Cognitive Tutors for math, which are in use by over 1300 school districts and by hundreds of thousands of students each year. After this notable success, however, most educational and training software is not of the ITS variety*."

Multiple factors could be impeding adoption, but two factors in particular stand out. The first is that ITSs are designed to be standalone systems that do not communicate or interoperate with other systems used to support learning, education, and training. The second is the sheer complexity of ITSs and the concomitant effort it takes to develop them (Murray, 2003).

With this as motivation, this chapter explores how standardization might help ITSs fit into learning ecosystems and simplify their design. Here we suggest focusing on standards for exchanging learner information among systems, *not* on standards for internal components. We also point out that the requirement to exchange learner information emphasizes the problem of determining which adaptations are responsible for the positive learning effect sizes observed when using ITSs. We believe that the suggested approach will enrich learner models, encourage developers to separate their innovative and proprietary adaptation engines from the portions of ITSs that interoperate with other learning systems, and ultimately, transform ITS architecture in ways that will make them easier to implement and adopt.

Standards

As Christensen & Raynor (2003) point out, interoperability and standardization enable competition and the growth of supply chains. In learning technology, for example, the emergence of learning management systems (LMSs) in the 1990s disrupted the print-based supply chain from authors to publishers to schools to learners. Standards such as IMS Content Packaging (IMS Global Learning Consortium, 2004), AICC Computer Managed Instruction (AICC, 2004), Sharable Content Object Reference Model (SCORM) 1.2 (Dodds, 2001), SCORM 2004 (ADL, 2006), and IMS Common Cartridge (IMS Global Learning Consortium, 2011) reestablished much of the same chain by allowing courseware to be produced independently and used by any compliant LMS. Arguably, the eLearning industry, which is a multi-billion dollar industry today (Adkins, 2011; Bersin, 2012; Global Industry Analysts, 2013), would not exist without these standards, and it is reasonable to assume that some standardization is needed to spur the adoption of ITSs.

At the same time, many proposed learning technology standards have failed to be completed or failed to achieve significant adoption. These proposals include standards for learner information and learner models, architectures, repository interoperability, intelligent agent communication, competency definitions, student identifiers, and many others. For example, the IEEE Learning Technology Standards Committee web site from December 2000 (IEEE, 2000a) shows 17 standards projects, almost all of which are still relevant today (including a standard for learner models). Less than half of these were completed in any form, and almost none have achieved any significant adoption. In most cases, failed efforts were driven more by an example and a vision rather than by a market/adopter pain point and real products. These failed standards, as described in Robson (2006), were "innovation and research-driven standards" rather than "market-driven standards." If standards are to be developed to spur the adoption of intelligent tutoring systems, they must be defined by market needs, or else they will have little effect.

Market Needs

What are those market needs? If we accept that ITSs are effective, the immediate market needs of potential customers are to (1) implement them with as little integration effort and work disruption as possible; and (2) obtain the data that are required for learning management and talent management and that can be used to demonstrate pedagogical impact and monitor costs. Customers may tolerate some inconvenience to get a more effective learning solution, but they are probably not willing to reconfigure their entire learning infrastructure, retrain all of their users, radically alter their established workflows, or give up on tracking grades and course completion status.

Unfortunately, ITSs make little or no attempt to exchange even basic results data with other systems. This isolationism is typical of new learning technologies. Early LMS and assessment engine products were the same, and so are massive open online courses (MOOCS) and the Khan Academy. The following exchange from the Google Khan Academy Developers Group (Azevedo & Ojeda, 11/19/12) typifies the reluctance of new product developers to address interoperability and the frustration of early adopters who typically run their daily operations via institutional learning management systems. (Typos and spelling corrected).

> DA: *Hello developers. Is Khan Academy / Khan Exercises SCORM Compliant?*
>
> MO: *hi D. Sorry to say, we do not. Is there some definite advantage to supporting it other than this graphic from scorm.com?" (Graphic shows reduced costs from using SCORM)*
>
> DA: *Hello and thanks for the quick response. I find one big use for SCORM. If a student uses multiple learning platforms it would be nice for grades and progress to be shared across the platforms. One example would be: I'm attending a class in Khan Academy, like Algebra II, but I find another site/course and I want to make Algebra III in that new site/course. If I could export/import my certified data across platforms that would be very flexible for students. In the long run people will like that the time spent on Khan or another site/course is certified and flexible, like in real universities there is equivalence in subjects and grades/progress.*[1]

The immediate market needs of potential customers are to track results and hold down implementation costs. These are conditions for diffusion of the technology, but once diffusion starts, other requirements will appear. For example, since ITSs provide individualized learning, it is likely that students will frequently switch among different systems. If the one system has gathered data about a learner's cognitive or affective characteristics, other systems can make use of it. Existing standards such as IMS GLC

[1] We note that one of the motivations for moving from SCORM to the Experience API is that SCORM does not address learners working on multiple platforms.

Learner Information Package (IMS Global Learning Consortium, 2005) and the Europass (Cedefop, 2012) can be used for transcript data, curriculum vitae (CV) data, and high level competencies (e.g., language skills), but new standards will be needed for expressing and comparing learner information.

In addition, adoption will bring renewed demands for avoiding lock-in. Even if the same content cannot be plugged and played in multiple intelligent tutoring systems, customers will want to leverage content they have already acquired and will want to outsource ITS development to multiple sources. To achieve this requirement, there must be some reasonable separation between the "system" software, "content" such as text and multimedia presented to the learner, and "interfaces," including the user interface and those that exchange data with other learning systems. This is diagrammatically shown in Figure 2-1.

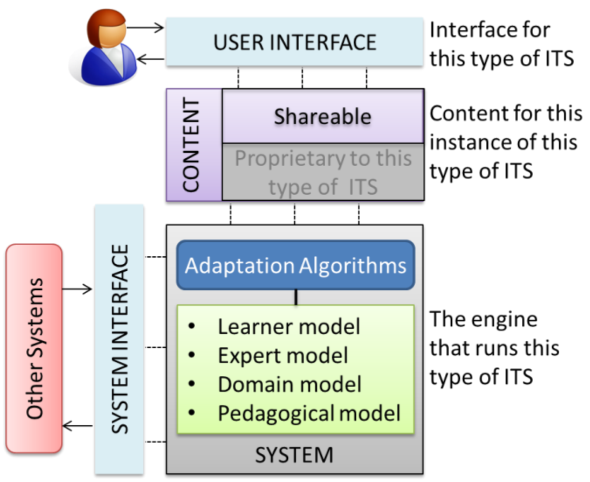

USER INTERFACE
Interface for this type of ITS
CONTENT
Shareable
Proprietary to this type of ITS
Content for this instance of this type of ITS
Other Systems
SYSTEM INTERFACE
Adaptation Algorithms
Learner model
Expert model
Domain model
Pedagogical model
SYSTEM
The engine that runs this type of ITS

Figure 2-1. System, Content and Interfaces

Standards for Learner Information

The first requirement for adoption is for ITSs to "play well" in current learning environments. This can be done by requiring conformance to SCORM and the Experience API or Tin Can API (Advanced Distributed Learning, 2013) or by adopting the IMS approach based on Learning Tools Interoperability (IMS Global Learning Consortium, 2012). The important point is that ITS developers should not ignore these standards if they want their systems to be adopted.

Requirements to exchange even very coarse data such as completion status and quiz results will naturally lead developers to reexamine their system architectures and, hopefully, lead to a separation of components such as that illustrated in Figure 2-1. For example, the Experience API, planned for the next generation of SCORM, uses an "actor – verb – activity" paradigm and is designed for compatibility with semantic inference engines (Poltrack, Hruska, Johnson & Haag, 2012). Using this application programming interface (API), a score on a quiz can be reported as a series of statements such as "Student completed quiz" and "Student scored 95." An inference engine embedded in an LMS or other learning system might additionally know that "Quiz assesses Quadratic Formula" and conclude that "Student demonstrates competency in Quadratic Formula." The requirement to generate such triples and support the API will suggest using a similar structure to store and exchange other data, including learner information. This may or may not be the optimal design choice for a particular ITS, but experience shows that developers of new systems often use standards as guidelines for functionality and design (Devedzic, Jovanovic & Gasevic, 2007).

Independent of how it is represented, the key question for standardization is what information should be exchanged. In other words, what should an ITS be telling other systems, including other ITSs, other enterprise learning systems, and other applications used by instructors, students, managers, and researchers? Since ITSs are valuable because of their positive effects on learning, this information should consist of the data responsible for attaining this effect. In other words, the question of what information should be exchanged is the question of what student data are required to achieve near optimal adaptation. This is a special case of the more general question, posed by Durlach (2012) and Ray Perez (personal communication), of what functionality in ITSs has the most effect on learning outcomes and how much of this functionality is needed in practice. We do not know the answer, but we can nonetheless make some reasonable conjectures concerning the types of learner information might be included:

- *Educational records and high level competencies such as language skills and flight certifications*. Standards exist for these data. It is not clear what inferences an intelligent tutor can make directly from them, but since human tutors find them valuable, there is an argument for including them.
- *Competencies (including Skills, Knowledge, Abilities, Outcomes, Objectives) and level of competence.* These have also been standardized and represent data that are crucial for talent management, job placement, and many other applications. They are also used for AICC/SCORM-type sequencing, and at a fine level of granularity, can be equated with domain topics.
- *Data in affective, motivational, and social dimensions.* Cognitive models are more common and better understood, but many adaptations rely on data in these dimensions (Dimitrova, 2009). For example, tutors may observe how students react to different types of stimuli and discover beliefs and attitudes that other systems could use to select instructional content and strategies.
- *Goals, including learning goals and mission/task goals.* These goals are also data that a human tutor would want to know and that might be important for adaptation.
- *Physical adaptations, such as location, device capabilities, ambient light, and accessibility data.* Accessibility data have been standardized, and these other data can clearly be used in many ways.

As an historical note, we believe that the task of standardizing learner data was first taken up by the IEEE Learning Technology Standards Committee. In 2000, the scope statement for the Learner Model working group (IEEE, 2000b) read: "*This standard will specify the syntax and semantics of a 'Learner Model,' which will characterize a learner (student or knowledge worker) and his or her knowledge/abilities. This will include elements such as knowledge (from coarse- to fine-grained), skills, abilities, learning styles, records, and personal information. This standard will allow these elements to be represented in multiple levels of granularity, from a coarse overview, down to the smallest conceivable sub-element. The standard will allow different views of the Learner Model (learner, teacher, parent, school, employer, etc.) and will substantially address issues of privacy and security*"

Its purpose consisted of five items:

1. *To enable learners (students or knowledge workers) of any age, background, location, means, or school/work situation to create and build a personal Learner Model, based on a national standard, which they can utilize throughout their education and work life.*
2. *To enable courseware developers to develop materials that will provide more personalized and effective instruction.*
3. *To provide educational researchers with a standardized and growing source of data.*
4. *To provide a foundation for the development of additional educational standards, and to do so from a student-centered learning focus.*
5. *To provide architectural guidance to education system designers.*

This project was known as "Personal and Private Information" (PAPI) and never turned into a standard for reasons beyond the scope of this chapter. It did, however, have some important attributes. It considered highly granular information, a consideration abandoned by the IMS Learner Information Package and Europass standards. It had personalized learning and architectural guidance as important use cases. Architectural guidance is important because the complexity of ITSs is a barrier to adoption. Interchange standards do not dictate how data are stored or processed internally, but they tend to influence how systems are designed. Devedzic et al. (2007), for example, give an extensive overview of eLearning standards aimed at developers of web-based eLearning systems because "*they establish high-level principles for organizing learning resources and developing web-based education (WBE) applications.*" As things stand, an ITS constructs its internal model of a specific learner solely from its interactions with that learner. The modeling framework is constructed from cognitive and learning science, but the only data used to instantiate the models come from interactions with the ITS. The ability to retrieve learner information from other systems will change that and encourage developers to separate their innovative and proprietary adaptation engines from the portions of their systems that interoperate with other learning technologies.

Models Should Not Be Standardized

We have suggested that learner information should be standardized but have avoided suggesting the same for *learner models*. In fact, we do not believe that standardization is appropriate for any of the models associated with the inner workings of ITSs.

In the abstract, ITSs adjust their instructional strategy based on a learner model, expert model domain model, and pedagogical model (Durlach & Ray, 2011), but very few real-world ITSs use all these components and the nature of each of these components can vary wildly. For example, the model-tracing cognitive tutors built by Carnegie Learning (Anderson, Corbett, Koedinger & Pelletier, 1995) encode expert knowledge in logical production rules and give hints to students when their input violates one of the rules, whereas ALEKS (2011) tracks which topics a student has mastered and, using data gathered from a large population of previous learners, infers which new topics the student is likely to be able to master next. In contrast to cognitive and model-tracing tutors, the AutoTutor family (Graesser et al., 2004; Hu et al., 2009) has less explicit encoding of domain knowledge. These tutors use semantic analysis to determine how relevant student input is to content that has been presented, and in some cases, adjust their strategies based on affective states determined by sensor data (D'Mello & Graesser, 2010; D'Mello & Graesser, 2012). An examination of examples cited in sources (Graesser, Jeon & Dufty, 2008; Murray, 2003; Ohlsson & Mitrović, 2006; Woolf, 2009) reveals even further diversity in the way that ITSs operate.

This diversity exists for a good reason. Motivated by the two-sigma effect size observed with one-on-one human tutoring, ITSs attempt to replicate this experience with technology (Kulik & Kulik, 1991; VanLehn, 2011). Each system's technical approach to building and using models to achieve this goal is its principal locus of innovation and a chief source of differentiation in the marketplace. There is a significant difference between macro-adaptations, which persist over time and can be used by multiple systems, and micro-adaptations, which are ephemeral and lie fully in the domain of a single system. The candidate standardization categories in the previous section are all macro-adaptations. Since ITSs derive their competitive advantage from micro-adaptations and the models that enable them, standardization of these models would not be accepted and, if accepted, may hinder innovation.

Recommendations for GIFT and Future Research

GIFT (Sottilare, 2012) provides a testbed in which multiple ITSs can function. Its architecture contemplates that ITSs will exchange data with an LMS. Adoption of GIFT would encourage ITS developers to include LMS-compatible reporting mechanisms, which we argue is a necessary step for market diffusion of ITS technology.

GIFT also includes a learner module, which we would view as learner information exchange service. There are existing standards that can serve as the basis for constructing such a service, and any reasonable candidate standard for learner information exchange will be extensible enough to evolve over time.

The key question, however, is what data ITSs actually need to exchange to enhance learning gains and support the other intelligent systems and apps used by teachers and students. This is a difficult research question, and the history of learning technology standardization teaches that abstract and theoretical answers to questions of this nature do not lead to practical and adoptable standards. The best approach is likely repeated cycles of experimentation and observation. That approach may require creating tutors that implement well-controlled and limited functionality, measuring their effect on learning, and iteratively refining standards for learner information exchange based on the results. A framework such as GIFT is ideally suited as the "breadboard" for such experimentation.

References

Adkins, S. (2011). The U.S. Market for Self-paced eLearning Products and Services: 2010-2015 Forecast and Aanalysis.

ADL. (2006). Sharable Content Object Reference Model (SCORM) 2004. (3rd ed.). Alexandria, VA: Advanced Distributed Learning initiative.

Advanced Distributed Learning. (2013). Experience API, from http://www.adlnet.gov/capabilities/tla/experience-api

AICC. (2004). CMI Guidelines for Interoperability (Rev 4.0 ed.): Aviation Industry CBT Committee.

ALEKS. (2011). ALEKS Home Page Retrieved October 20, 2011, from http://www.aleks.com

Anderson, J. R., Corbett, A. T., Koedinger, K. R. & Pelletier, R. (1995). Cognitive tutors: Lessons learned. *The journal of the learning sciences, 4*(2), 167-207.

Azevedo, D. & Ojeda, M. (11/19/12). SCORM, from https://groups.google.com/forum/?fromgroups=#!msg/khanacademy-developers/pEts3twbk5g/3Y9SL4sHgZoJ

Barr, A., Beard, M. & Atkinson, R. C. (1975). A rationale and description of a CAI program to teach the BASIC programming language. *Instructional Science, 4*(1), 1-31.

Bersin, J. (2012). LMS 2013: The $1.9 Billion Market for Learning Management Systems: Deloitte.

Blessing, S. B., Gilbert, S. B., Ourada, S. & Ritter, S. (2009). Authoring model-tracing cognitive tutors. *International Journal of Artificial Intelligence in Education, 19*(2), 189-210.

Cedefop. (2012). Europass Retrieved February, 2013, from http://europass.cedefop.europa.eu/en/home

Christensen, C. M. & Raynor, M. E. (2003). *The innovator's solution: Creating and sustaining successful growth*: Harvard Business Press.

Devedzic, V., Jovanovic, J. & Gasevic, D. (2007). The pragmatics of current e-learning standards. *Internet Computing, IEEE, 11*(3), 19-27.

Dimitrova, V. (2009). *Artificial Intelligence in Education: Building Learning Systems that Care: From Knowledge Representation to Affective Modelling* (Vol. 200): Ios PressInc.

Dodds, P. (2001). Sharable Content Object Reference Model v 1.2: Advanced Distributed Learning Initiative.

Dodds, P. & Fletcher, J. D. (2004). Opportunities for New "Smart" Learning Environments Enabled by Next-Generation Web Capabilities. *Journal of Educational Multimedia and Hypermedia, 13*(4), 391-404.

Durlach, P. J. (2012). Vanilla, Chocolate, or Chunky Monkey: Flavors of Adaptation in Instructional Technology. *iFest 2012*, from http://www.adlnet.gov/wp-content/uploads/2012/08/Durlach_Adaption_in_IT_iFest-2012.pdf

Durlach, P. J. & Ray, J. M. (2011). Designing adaptive instructional environments: Insights from empirical evidence *Army Research Institute Report*. Arlington, VA.

Global Industry Analysts. (2013). Global eLearning Market to Reach $107.3 Billion by 2015. Reported on PRWeb., from http://www.prweb.com/releases/elearning/corporate_elearning/prweb4531974.htm

Graesser, A. C., Jeon, M. & Dufty, D. (2008). Agent technologies designed to facilitate interactive knowledge construction. *Discourse Processes, 45*(4-5), 298-322.

Graesser, A. C., Lu, S., Jackson, G. T., Mitchell, H. H., Ventura, M., Olney, A. & Louwerse, M. M. (2004). AutoTutor: A tutor with dialogue in natural language. *Behavior Research Methods, 36*(2), 180-192.

Hu, X., Cai, Z., Han, L., Craig, S. D., Wang, T. & Graesser, A. C. (2009). *AutoTutor Lite*.

IEEE. (2000a). IEEE Learning Technology Standards Committee (LTSC) Retrieved February, 2013, from http://web.archive.org/web/20010217054051/http://grouper.ieee.org/LTSC/

IEEE. (2000b). IEEE P1484.2 Learner Model Working Group Retrieved February, 2013, from http://web.archive.org/web/20010221110739/http://ltsc.ieee.org/wg2/index.html

IMS Global Learning Consortium. (2004). IMS content packaging specification v 1.1.4.

IMS Global Learning Consortium. (2005). IMS Learner Information Package.

IMS Global Learning Consortium. (2011). IMS Common Cartridge Specification.

IMS Global Learning Consortium. (2012). Learning Tools Interoperability, from http://www.imsglobal.org/toolsinteroperability2.cfm

Kulik, C.-L. C. & Kulik, J. A. (1991). Effectiveness of computer-based instruction: An updated analysis. *Computers in human behavior, 7*(1), 75-94.

Murray, T. (2003). An Overview of Intelligent Tutoring System Authoring Tools: Updated analysis of the state of the art. Chapter 17 in Murray, T., Blessing, S. & Ainsworth, S. *Artificial Intelligence*.

Ohlsson, S. & Mitrović, A. (2006). Constraint-based knowledge representation for individualized instruction *Computer Science and Information Systems, 3*(1), 1-22. doi: 10.2298/CSIS0601001S

Poltrack, J., Hruska, N., Johnson, A. & Haag, J. (2012). *The Next Generation of SCORM: Innovation for the Global Force.* Paper presented at the The Interservice/Industry Training, Simulation & Education Conference (I/ITSEC).

Robson, R. (2006). Globalization and the future of standardization. *Computer, 39*(7), 82-84. doi: 10.1109/mc.2006.231

Sleeman, D. & Brown, J. S. (1982). Intelligent tutoring systems.

Sottilare, R.A., Brawner, K.W., Goldberg, B.S. & Holden, H.K. (2012). *The Generalized Intelligent Framework for Tutoring (GIFT)*. Orlando, FL: U.S. Army Research Laboratory – Human Research & Engineering Directorate (ARL-HRED).

VanLehn, K. (2011). The Relative Effectiveness of Human Tutoring, Intelligent Tutoring Systems, and Other Tutoring Systems. *Educational Psychologist, 46*(4), 197-221.

Woolf, B. P. (2009). *Building intelligent interactive tutors: Student-centered strategies for revolutionizing e-learning*. Burlington, MA: Morgan Kaufmann.

CHAPTER 3 – Important Considerations for Learner Models: Transfer Potential and Pedagogical Content Knowledge

Alan Lesgold[1] and Arthur Graesser[2]
[1] University of Pittsburgh; [2] University of Memphis

Introduction

In the age of the intelligent machine, training of humans inevitably requires training for transfer. If we knew exactly what a person should do in a situation, we could program a computer or robot to do it instead. However, no two situations are exactly the same so uncertainty and differences arise in matches between the current situation and episodes of the past. Modeling transfer has traditionally meant getting as close as possible to modeling the total expertise trainees were expected to have in handling episodes of the past and future. We suggest that there may be more effective approaches today. One possibility is to model knowledge (general or specific) that would permit a person to represent a situation in a form that can be handled by diverse training and transfer trajectories. A second possibility would be to establish a principled way to prioritize which knowledge should be overlearned so that it can be “stretched” to novel situations not covered in training. A related approach would be to do both. For example, one could perform situation modeling within the domain of interest along with cognitive load analyses of the situation modeling examples drawn from the target domain. This chapter explores these options and considers what task analysis and expert modeling is needed to capture them.

If we want people to carry out performance of a task in a precise way, we can conduct a task analysis and then teach each of the elements of the performance and directly measure how well each element is acquired. This approach to training worked remarkably well for the training of line workers in the industrial age and of enlisted personnel in the military prior to the knowledge revolution (Collins & Halverson, 2009). It assumed availability of supervisors or officers who would creatively handle unexpected challenges and provide direction that allowed teams to adapt to emergent situations. It was generally assumed that these leaders did not need carefully monitored complete training for their roles but rather that they would be selected as being intelligent enough to prove a useful bridge from the formal training of their subordinates to the actual situations their unit might encounter.

Now that we are entrenched in the knowledge revolution, however, routine performance can be specified so precisely that we can teach a person or program an intelligent machine to perform these procedures competently. Moreover, once empowered by intelligent tools, every worker or soldier has a role akin to the leaders or officers of times past. Nevertheless, experience has shown that most workers in intelligent work environments need substantial training, even if we cannot fully drill them on every aspect of the performances we hope they will exhibit. This creates a need to consider transfer in the design of training. That is, we are no longer merely preparing people by teaching them all the elements, rules, or procedures that define perfect performance in every likely circumstance. Rather, they need to be prepared for emergent situations that deviate from original training. Our hope is that our trainees will show transfer from the specific prescribed training to ideal performance on emergent tasks.

Related Research

The profound challenge is that there typically is unspectacular transfer from training to new situations (Banich & Caccamise, 2010). A classical study by Hayes and Simon (1977) had college students attempting to solve a series of problems that had structurally identical solutions but varied in surface characteristics, such as substituting names of characters and objects (Hobbits and Orcs vs. Monsters and

Globes). Transfer between four successive problems was near zero unless students were explicitly instructed to make similarity connections between the problems. Gick and Holyoak (1980) investigated whether the structure of a story about troops converging on a location would help college students solve a radiation problem that had a direct structural isomorph to the story. The transfer was modest without cognitive activities that intentionally tried to establish correspondences between the two representations. At a more global practical level, the order of courses in a curriculum is rarely backed by substantiating transfer data (Vuong, Nixon & Towle, 2011). For example, there is not an abundant body of empirical evidence that calculus helps engineering or that physics helps students understand chemistry. Knowledge and skills are highly constrained by specific characteristics of the subject matter.

The above examples present a dismal picture of transfer, but it can be countered by exemplars of successful transfer. The literature is replete with evidence that training on particular tasks can facilitate performance on similar tasks in the future. The devil is in the details of the similarity of the stimuli, tasks, and associated cognitive representations (Gentner & Markman, 1997).

The transfer problem has enormous implications for the GIFT architecture (Sottilare, Brawner, Goldberg & Holden, 2012), particularly with respect to the distinction between domain-independent and domain-dependent modules. The current GIFT architecture specifies that the sensor and pedagogical modules are domain-independent whereas the subject-matter knowledge is domain dependent. The implication of the domain-independent modules needs to be clarified. Does that mean that there should be significant transfer between tasks that have similar subject matters but different sensor and pedagogical modules? Does that mean that the same pedagogical methods can be applied to a broad set of subject matters, as opposed to the pedagogical modules being distinctively tailored for particular subject matters? Does subject matter (domain knowledge) reign supreme over sensor and pedagogical modules? What are the priorities among these GIFT modules on the matter of transfer?

We propose that a model of student[2] learning has two categories of knowledge in an intelligent training system. First, it knows how well the student has mastered the elements specifically being trained. Second, it knows how prepared the student is to confront categories of situations that cannot be predicted and cannot be rehearsed adequately. This latter requirement means that such systems must embody a theory of transfer that can allow us to know how far a student's knowledge might stretch.

Consider, for example, a football coach. The coach might train the team to execute a particular kind of play, such as an option running play. The team can practice each of the two or three ball carrier options, so that every team member knows who to pass the ball to, who to get the ball from, or where the players position themselves to block the defense. However, the practice sessions only rehearse some of the possibilities because the defensive players are intelligent entities and do not necessarily behave exactly as the sham defense set up during the practice sessions. Yet the coach is hopeful that the team will perform well in every case.

Some of the performance environments are predictable. By having the defense behave intelligently and conform to known football best practices, the coach can create a variety of practice situations that anticipate what will happen in an upcoming game. However, the game will have situations that deviate from the ones on which practice occurred. A theory of practice is useful to the extent that it allows the coach to predict game performance from practice performance, select the right practice situations, and decide when there has been enough practice.

[2] For simplicity of exposition, we use terms like "student model" and "students" even though the primary audience for this volume are training developers and training system designers, not schoolteachers or professors.

Unpredictable situations will nevertheless occur. It might be icy or rainy. The field might be muddy or not the same surface as that on which practice occurred. The players may change as wounds heal, injuries occur, and so on. Consequently, there need to be global indicators of whether team members are ready for a wider range of unpredictable occurrences. A decision is needed as to whether the current situation is under the realm of status quo or an unusual case. An adequate model would identify specific performances that a player can do well in prototypical situations as well as some estimates of which categories of unpredictable cases can be expected to be adequately handled. The latter models the transfer potential of what was explicitly learned. One way to address the problem of transfer is to consider the two skill sets separately. That is, there are the specific performance skills that encompass expertise in the target domain. There are also *situation representation* capabilities that allow a student to classify a novel situation as one in which particular learned performances can be successful. These situation representational skills mediate the availability of learned rules to be applied in emergent situations. Our experiences in building intelligent training systems have convinced us that these two skill sets should be handled in different ways.

The first author has developed and tested intelligent training systems for maintaining equipment in the military and business sectors, including troubleshooting equipment failures (Gott & Lesgold, 2000; Gott, Lesgold & Kane, 1997; Lesgold & Nahemow, 2001). For the specific performance skills, there was the standard approach of building an expert model that was capable of solving the range of tasks that was targeted for training. The initial assumption was that once each rule in the expert model was demonstrated by the student, training was considered completed. However, our experience quickly revealed that this was not entirely the case because the target domains we addressed were extremely complex. All of the information needed to trigger each of the rules relevant for a given task was present in the task domain, but there was so much information that a serious challenge remained in representing the situation at hand sufficiently to trigger appropriate rules.

The challenge of system complexity required us to train the situation representation skills needed to perceive the problem domain adequately to trigger expert rules. Our expert informants initially believed that this required training on recognizing the various components of the complex target domain. For example, in the case of avionics test stations, this would include the system modules and the components of the modules. However, it was apparent that all of the students could recognize all of the modules and their components, even at the beginning of training. The needed situation representation skills were more subtle than our expert informants realized.

A reasonable training method would be to directly teach how to recognize which aspects of a situation are important. This is done in football when players are taught the names and overall strategies behind various types of plays. It also is done in medicine when physicians in training directly learn to recognize various syndromes for which diagnostic rule sets are known. This approach to teaching for transfer can work when there are recurrent examples of these various plays or syndromes. In football, that happens because the coach has studied game films and knows the patterns that the opposing team is likely to use. In medicine, it happens because various genetic and environmental factors predispose human bodies to exhibit various syndromes or patterns of malfunction (e.g., lots of us overeat and under-exercise, with the bodily response being pretty stereotypic).

Engineered systems generally are built and modified to adapt to the range of circumstances in which they are deployed, but unfortunately they often do not have predictable patterns of breakdown. If those existed, they would be engineered away. This makes it harder to teach situation representation by teaching how to recognize syndromes; there are few if any recurrent syndromes to teach. This required us to take a different approach that did not explicitly teach students to watch for specific patterns, at least for the most part. Instead, students were provided a useful *range* of experiences that prompted them to construct rule-based knowledge on top of their prior experiences with the task domain. This more flexible emphasis on the range of the situations was expected to extend the students' knowledge to new situations.

The scheme that was developed was called *intelligent coached apprenticeship*. Basically, we pushed students to go beyond their overlearned knowledge to solve novel problems (Gott & Lesgold, 2000). We expected that none of the problem tasks our training system provided would be solvable by the students without help. Five levels of help were available. The lowest level listed the sequence of actions the student already had taken to address the problem. More than half of the time, that level of hint was sufficient to keep the student moving toward a solution to the problem. If the student was still stuck, there would be more explicit levels of hint (e.g., where to look for information) until a final hint level that told the student exactly what to do on the next step. For the population our training addressed, the desire to learn was strong, and students seldom asked for more help than they needed. After completing the solution for the problem, they also were able to study a comparison of their solution path to that of an expert.

The central hypothesis in this research (Gott & Lesgold, 2000; Gott, Lesgold & Kane, 1997) is that scaffolding for students to shape their own construction of situation representation skills would produce the needed development of those skills, even if we could not explicitly list them all and even if it was not feasible to build a complete set of situation representation rules and train on each. It was reassuring that our efforts were highly successful. We had expert technicians develop a collection of far transfer tasks that involved applying the expert rules we taught to new hardware. The transfer hardware was a fictitious piece of hardware that could still be diagnosed using the expert rule set that our training system embodied. The students we trained performed in about the same range as experts on those problems (Gott & Lesgold, 2000; Gott, Lesgold & Kane, 1997).

It may eventually be possible to teach the needed situation representation skills directly rather than to assume them and indirectly stimulate their construction by the students. For example, Forbus et al. (2007) have made progress in specifying what a computer needs in order to learn similar kinds of representational skills. Nevertheless, the scheme we developed is worthy of further exploitation. On the one hand, it focuses on the expert rules needed to do the necessary problem solving, and on the other hand, it does not ignore the reality that some level of further knowledge construction is needed if those rules are to be available when needed in addressing novel tasks.

There will always be uncertainty that an individual student's understanding of the task domain embodies entirely the same constructs that the training designer might have had. This is especially the case in complex technical domains, where the training designers often know much more basic science than the technicians being trained. As an example, consider the terms used in expert rules for diagnosing failures of an ion beam system for writing circuits on computer chips (Lesgold & Nahemow, 2001). A complete account requires knowledge of quantum physics, silicon chemistry, and optics. Technicians two years out of high school could learn the diagnosis rules well enough to apply them in transfer situations using the training technique of providing difficult problems with scaffolding and post-problem reflection (Lesgold & Nahemow, 2001).

Discussion

The intelligent coached apprenticeship was impressively effective, producing learning effect sizes exceeding one standard deviation. These results are on par with or exceed intelligent tutoring systems that have been developed and tested during the last decade (Graesser, Conley & Olney, 2012; VanLehn, 2011). However, questions remain about the information that is needed in the students' situation representation to assure adequate learning. There are also questions directly relevant to GIFT (Sottilare et al., 2012).

The systems described in the previous section did not log information about the students' situation representation, and it did not retain its short-term estimations of student reasoning strategies. Instead, the systems were validated by conducting far transfer studies. This is an expensive way to proceed. It is worth considering simpler ways to assure adequate learning of rules that provide substantial transfer. Some of these possibilities are presented in this section.

One important step would be to record information about the level of coaching/scaffolding that a student requires. The systems described above did that, but the data were not retained beyond their presence in a post-problem recapitulation of the student's activity and a comparison of that to an expert solution. GIFT, in contrast, routinely records in the log files the raw sensor data, computer-student interactions, pedagogical strategies being implemented, expected student responses (both correct and misconceptions), actual student responses, and other data that can be mined for system improvement. At any given moment in solving a problem in the intelligent coached apprenticeship systems, the system kept track of what the best next step for a student. Therefore, it would be straightforward to record data that indicates that the student has a misconception or does not know what to do next (see, for example, the inference of "malrules" by systems VanLehn [1981] has built). Informative episodes occur when the student asks for coaching or performs a non-optimal next step. As the student starts taking appropriate actions, the past records of incomplete learning could be retired and eventually there would be confidence that a rule has been learned broadly. GIFT supports such data mining and machine learning activities.

Note that the approach is substantially different from direct training of a rule. In direct training, the student is placed in circumstances where the rule should be triggered and then taught explicitly to apply it. The problem with the direct training approach is that the rules flagged as learned may not be triggered in circumstances where complex tasks are being performed without attention focused directly on possible circumstances where the rule applies. We have known for almost a century (Whitehead, 1929) that specifically learned bits of knowledge often are not used in broader circumstances where they should apply. By focusing more of learning on explicitly stretching one's knowledge, this problem has been avoided. The question arises whether training of humans is needed at all, given that the intelligent coached apprenticeship systems relied upon expert models to provide coaching. Expert models would perhaps not be able to transfer to novel situations any more than novice technicians. We were able to formally represent each problem in a way that permitted the expert system to solve it, but those representations were only possible because of the additional knowledge of the experts who developed them. That more implicit knowledge is exactly what intelligent coached apprenticeship endeavors to train.

Decisions will need to be made on how the above mechanisms in the intelligent coached apprenticeship would be implemented in GIFT. The student-constructed situation representation is not merely stored in the GIFT Domain Module, but rather is apparently derived in the Learner Module from a combination of activities involving the Sensor Module, Pedagogical Module, the Domain Module, and history of the logged data. Does this complex interactivity clash with the modular assumption that differentiates the domain-dependent Domain Module from the remaining domain-independent components?

Another important step for implementing effective intelligent coached apprenticeship systems resides in tracking mastery of rules in broad contexts. The problem sets presented to students need to be sufficiently challenging and span a wide enough range of situations. They can never span all the situations a student will encounter when applying what is being taught. However, they need to span a sufficiently wide range that they force students to reflect on why each rule is applicable and the range of possible situations of applicability. For the two rather different domains developed by Lesgold and his colleagues, the expert technicians helped build problems that are sufficiently diverse that there was transfer to novel situations.

One situation involved diagnosis of failures in complex switching systems designed to test electronic components from military aircraft. The range of problems experts helped us develop was sufficient to produce far transfer performance that was about as good as that of domain experts (Gott, Kane & Lesgold, 1995; Gott, Lesgold & Kane, 1997). The second domain in which we did similar training and testing with similar results was the diagnosis of failures in machines used to make computer chips, notably ion deposition systems that put layers on chips and ion beam implant systems that write circuits on those layers (Lesgold & Nahemow, 2001). The switching systems had underlying knowledge associated with simple electrical properties of circuit continuity and magnetics (relays were involved). The chip-making systems involved underlying knowledge of electrical systems, heat distribution, silicon gas compounds, movements by robots, and to some extent, even more complex basic science.

The same basic approach worked for both domains so it is at least a reasonable conjecture that it can work much more widely. It would be worthwhile, however, to study the utility of logging indications of uncertainty, lack of knowledge, or misconceptions at points where the rules in an expert system are not applied routinely or accurately. Such information might allow better decisions on how much training is needed to produce mastery. Once again, the logged data, data mining, and machine learning facilities of GIFT are, in principle, equipped to implement this approach. Decisions will need to be made on the division of labor among GIFT modules when accommodating changes to the system.

Recommendations and Future Research

The intelligent coached apprenticeship system described in this chapter underscores the importance of capturing situation representations that are constructed by the students in order to handle a diverse range of cases applicable to transfer situations. Domain experts will be needed to select the problems that deviate from routine cases that can be trained explicitly. Research will therefore be needed to understand how these cases/problems are selected, the mapping between training and transfer cases, the situation representations that students construct, and their ability to identify unusual cases. A detailed analysis of the log data should be helpful in these research efforts and also in modification of the systems in iterative development.

There are two major recommendations directly relevant to GIFT. First, decisions will need to be made on how the specific modules will participate in the intelligent coached apprenticeship system. It is not a simple matter of storing content in the Domain Module. Second, decisions will need to be made on how to represent the information stored in the log files and the various modules. It is not a simple matter of storing everything. The features, content, and structures will need to be able to support new cases and situation representations in addition to domain-dependent and domain-independent information.

References

Banich, M.T. & Caccamise, D. (2010)(Eds.) Generalization of knowledge: Multidisciplinary perspectives. New York: Psychology Press. .

Collins, A. & Halverson, R. (2009). Rethinking education in the age of technology: The digital revolution and schooling in America. New York: Teacher College Press.

Forbus, K., Riesbeck, C., Birnbaum, L., Livingston, K., Sharma, A. & Ureel, L. (2007). Integrating natural language, knowledge representation and reasoning, and analogical processing to learn by reading. Proceedings of AAAI-07: Twenty-second Conference on Artificial Intelligence. (1542-1547). Vancouver, BC: AAAI.

Gentner, D. & Markman, A. (1997). Structure mapping in analogy and similarity. American Psychologist, 52, 45-56.

Gick, M.L. & Holyoak, K.J. (1980). Analogical problem solving. Cognitive Psychology, 12, 306-355.

Gott, S. P., Kane, R. S. & Lesgold, A. (1995, February). Tutoring for transfer of technical competence. Armstrong Laboratory Technical Report AL/HR-TP-1995-0002. Brooks AFB, TX: Air Force Materiel Command.

Gott, S. P., Lesgold, A. M. & Kane, R. S. (1997). Promoting the transfer of technical competency. In S. Dijkstra, et al. (Eds.), Instructional design: International perspectives (vol. 2). (221-250). Hillsdale, NJ: Erlbaum.

Gott, S. P. & Lesgold, A. M. (2000). Competence in the workplace: How cognitive performance models and situated instruction can accelerate skill acquisition. In R. Glaser (Ed.), Advances in instructional psychology. (239-327). Hillsdale, NJ: Erlbaum.

Graesser, A.C., Conley, M. & Olney, A. (2012). Intelligent tutoring systems. In K.R. Harris, S. Graham, and T. Urdan (Eds.), APA Educational Psychology Handbook: Vol. 3. Applications to Learning and Teaching (pp. 451-473). Washington, DC: American Psychological Association.

Hayes, J.R. & Simon, H.A. (1977). Psychological differences among problem isomorphs. In J. Castellan, D.B. Pisoni & G. Potts (Eds.), Cognitive theory, vol. 2. Hillsdale, NJ: Erlbaum.

Lesgold, A. & Nahemow, M. (2001). Tools to assist learning by doing: Achieving and assessing efficient technology for learning. In D. Klahr & S. Carver (Eds.), Cognition and instruction: Twenty-five years of progress. Mahwah, NJ: Erlbaum.

Sottilare, R.A., Brawner, K.W., Goldberg, B.S. & Holden, H.K. (2012). *The Generalized Intelligent Framework for Tutoring (GIFT)*. Orlando, FL: U.S. Army Research Laboratory – Human Research & Engineering Directorate (ARL-HRED).

VanLehn, K. (1981). Bugs are not enough (Tech. Rep. No. CIS-11). Palo Alto, CA: Xerox Research Center.

VanLehn, K. (2011). The relative effectiveness of human tutoring, intelligent tutoring systems and other tutoring systems. Educational Psychologist, 46, 4, 197-221.

Vuong, A., Nixon, T. & Towle, B. (2011). A method for finding prerequisites within a curriculum. In Pechenizkiy, M., Calders, T., Conati, C., Ventura, S., Romero , C., and Stamper, J. (Eds.), Proceedings of the 4th International Conference on Educational Data Mining (pp. 211-216).

Whitehead, A. N. (1929/1967). The aims of education and other essays. New York: The Free Press.

CHAPTER 4 –Matching Learner Models to Instructional Strategies

Andrew M. Olney and Whitney L. Cade
University of Memphis

Introduction

Learner models represent key variables that guide instructional strategies during e-Learning. By representing key variables about a particular student, learner models make adaptive instruction possible. Without learner models, instruction cannot be individualized and instead is often calibrated to the average ability learner or the lowest ability learner. Learner models enable instruction to focus on what the student doesn't know and adjust to the student's abilities, motivation, and preferences. In other words, learner models enable adaptive instructional strategies.

Given the tight correspondence between learner models and instructional strategies, they must be considered in parallel when designing e-Learning systems. In fact, a strong argument could be made that the choice of instructional strategy drives all other modeling decisions in e-Learning by placing requirements on the learner model and other related models (see Pavlik et al., Chapter 5 in this volume, for a review). Thus, one approach to studying learner models would be to conduct an analysis of all possible instructional strategies and then examine what requirements they place on learner models. However, the vast number of instructional strategies that have been proposed in the literature make this approach somewhat impractical.

In this chapter, we explore an alternative approach to strategy-based analysis of learner models. The Institute for Simulation and Training at the University of Central Florida has recently published an online database known as the Instructional Strategies Indicator (ISI) (Tarr, 2012). The ISI contains 150 instructional strategies, indexed by when and where instruction occurs, the evidence for instructional efficacy, the size of the group being instructed, the expertise of the learner, and the type of knowledge being taught. The ISI represents an effort to organize the known instructional strategies into a comprehensive framework, allowing for the optimal selection of an instructional strategy in a given instructional setting.

The methodology used to create the ISI was qualitative and data-driven, using aspects of grounded theory methodology to select relevant literature and create an analytic framework to describe the literature. Vogel-Walcutt, Fiorella, and Malone (2012) conducted searches of PsychInfo, the Educational Resource Information Center (ERIC), and Google Scholar using a predefined set of search terms and restricting the dates of studies to between 2000 and 2010. Of the 4,515 articles returned, only 771 were retained as relevant to the ISI criteria, which included relevance to military training. In addition to being coded according to the dimensions mentioned above (e.g., when and where instruction occurs), instructional strategies in the retained articles were rated by judges on their associated evidence for efficacy. Strategies were ranked on a 0–9 scale based on multiple criteria of evidence, including empirical results and quality of study, with judges' ratings being checked for inter-rater reliability (Vogel-Walcutt, Malone & Fiorella, 2012). This process yielded 150 different instructional strategies that were included in the ISI. However, only 13 of these strategies were given the highest rating of 7–9, which was reserved for strategies backed by multiple randomized experiments with moderate to large effect sizes ($d \geq 0.5$; Cohen, 1992).

Table 1 presents these 13 strategies with a subset of the ISI dimensions. Missing dimensions include hierarchical categorizations of the strategy type and the knowledge, skills, and abilities to be learned. Included dimensions are (1) timing of instruction relative to the instructional event (pre/during/post), (2) setting of instruction (e.g., computer-based, classroom, or live-training), (3) group size (e.g., individual, small group, or large group), (4) learner's level of expertise (novice/journeyman/expert), (5) knowledge type targeted (declarative/procedural/conceptual/integrated), and (6) rating of evidence for efficacy. Each of these dimensions has previously been described in detail (Vogel-Walcutt, Fiorella & Malone, 2012), but it is outside the scope of this chapter to evaluate the ISI framework or methodology. For the present purposes, it is important to note that the ISI represents a serious attempt to frame the landscape of instructional strategies, and as such it has implications for the study of learner models. The purpose of this chapter is to consider the alignment of learner models to the instructional strategies in Table 4-1.

Table 4-1. Top 13 instructional strategies from the ISI

Name	Time	Setting	Group Size	Level	Knowledge	Rating
Goal Setting	Pre	Any	Individual	Novice	Declarative	7
Scaffolding	During	Any	Individual	Novice	Declarative	8
Distributed Practice	During	Any	Individual	Novice	Declarative	8
Massed Practice	During	Any	Individual	Novice	Procedural	8
Adaptive Instruction	During	Classroom	Individual	Novice	Conceptual	8
Multimedia Instruction	During	Computer-based	Individual	Novice	Conceptual	8
Intelligent Tutoring Systems (ITS)	During	Computer-based	Individual	Novice	Integrated	8
Affective ITS	During	Computer-based	Individual	Journeyman	Conceptual	7
Bayesian Approach	During	Computer-based	Individual	Novice	Conceptual	7
Peer Learning	During	Classroom	Large group	Novice	Conceptual	7
Pedagogical Agent	During	Computer-based	Individual	Novice	Integrative	7
Self Reflection	Post	Classroom	Individual	Novice	Conceptual	7
Self Assessment	Post	Classroom	Individual	Journeyman	Integrative	7

However, even a quick inspection of Table 4-1 reveals that these "strategies" are not equally comparable. Adaptive Instruction is properly viewed as a category that includes ITS, affective ITS, Scaffolding, and the Bayesian Approach, so it arguably could be excluded from further discussion. Likewise, ITSs are a vehicle for delivering strategies; there is no a priori strategy associated with ITSs, except that they are somehow individually adaptive. Finally, the Bayesian Approach is a specific modeling formalism, not a strategy. Although some of the others are not well-defined "strategies" in a strong sense, they are identifiable with strategies, such as Multimedia Instruction (optimal mode of presentation), Pedagogical Agent (social enhancement of learning), and Affective ITS (learning enhancing affect). Therefore, only the following ten strategies will be considered with respect to learner models in the remainder of this chapter. These ten strategies form three natural groupings of self-regulated learning strategies (Goal

Setting, Self-Assessment, and Self-Reflection), social constructivist strategies (Scaffolding, Affective ITS, Peer Learning, and Pedagogical Agent), and memory-enhancing presentation strategies (Distributed Practice, Massed Practice, and Multimedia Instruction). We discuss each of these in turn.

Self-Regulated Learning Strategies

Three of the above ISI strategies are aligned with the four phases of self-regulation: planning/goal setting, monitoring, control, and reflection (Pintrich, 2000). As we proceed, it is important to note that, in a tutor-student or teacher-classroom context, these four phases may be enacted in a distributed way. For example, while the student may have set the initial learning goals, the tutor may be monitoring the student's progress and controlling strategies to facilitate the student's progress. E-learning environments likewise have the same potential to distribute and scaffold self-regulation strategies.

The four phases of self-regulation may manifest in several ways. At any point during the learning session, learners may set goals for learning or goals for performance. During the learning session, learners may monitor their progress toward goals (e.g., judgments of learning or number of pages read). Depending on perceived progress towards goals, learners may invoke a number of control strategies (e.g., paraphrasing, summarizing, note-taking). Finally, learners may reflect on their performance in a more holistic sense and make corresponding attributions (e.g., the material is difficult so this took a long time). Although the four phases above were described from a cognitive viewpoint, the same phases can be applied to self-regulation of motivation/affect, behavior, and learning context (Pintrich, 2000). The following sections describe these four phases in more detail along with the learner model variables needed to support them.

Goal Setting

Reviews of goal setting have found that the specificity, proximity, and difficulty level of goals influence the effects of goal setting (Schunk, 1990; Locke & Latham, 2002). Expended effort increases linearly with task difficulty until the limits of ability are reached. Moreover, specific and difficult goals lead to higher performance than general goals like "Do your best." While general goals reference no objective criterion and so may be ambiguous, specific goals allow a better determination of the required effort and whether success has been achieved. Proximal goals likewise lead to better performance, because progress towards proximal goals is easier to determine than it is for distal goals.

Four mechanisms have been identified that explain the effect of goals on performance (Locke & Latham, 2002). First, goals focus attention of goal-relevant activities. This can help learners ignore information that is irrelevant to the goal. Second, goals activate relevant schemas and procedures for attaining the goal. In cases where the task is familiar, planning and execution will be largely automatic. Third, as mentioned above, level of effort is directly proportional to perceived difficulty. Therefore, more difficult goals can lead to greater effort expended. Fourth, in tandem with increased effort, difficult goals tend to increase the amount of time spent on a task. Given these basic mechanisms involved with goal setting, it makes sense to design instruction so that the performance of the learner is maximized. A fuller description of variables that interact with these mechanisms is reviewed by Locke and Latham (2002).

With these basic principles in mind, it is straightforward to outline some capabilities for a learner model that supports goal setting. First, the learner model should support setting goals before the session that are specific, difficult, and proximal. Thus, ideally, the learner model should not just be used to store arbitrary goals in memory, but should also be used to assess these goals with respect to the student's ability level and the nature of the domain. To prevent goals that are vague, too easy/too difficult for the student, or too distant for the learning session, the learner model should support both evaluation of goals and adaptive guidance to give students feedback on goals. For example, if the goal is too vague, the system might say, "That seems a little vague. How about a more specific goal?" or if the goal is too difficult, the system might say, "That's a good long-term goal, but why don't you simplify it to make it achievable in the amount of time you have?" Of course, assessment of the dimensions of specificity, difficulty, and proximity require complementary modeling of the domain, and the specific feedback offered requires complementary pedagogical modeling (though of a relatively limited sort). Furthermore, if the learner model contains longitudinal information about the student and their previous goals, this information could also be applied to the current situation. For example, if the current situation were similar to a previous situation, the system could remind the user of the associated goal from that previous situation.

Self-Assessment

Self-assessment, by which some criterion (e.g., a goal) is used as a reference to determine progress, is central to monitoring in self-regulated learning. Criteria range from task-specific, as discussed above, to general goal orientations (e.g., mastery and performance goals) (Pintrich, 2000). Externally supplied feedback that link progress towards goals to student strategy use has been found in multiple studies to promote skill and self-efficacy, both when the product of learning and the process of learning are emphasized (Schunk, 1990). Self-assessments may also supply feedback by evaluating level of understanding, personal interests, effort, strategies, history of improvement, and perceived strengths and weaknesses, while fostering students' sense of control over their learning (Paris & Paris, 2001). In a review of the principles of self-assessment, Paris and Winograd (1999) identify three ways in which self-assessment can improve learning. First, awareness of different ways of learning increases when the learning styles and strategies of others are compared to those of the self. Second, efficient allocation of attention and effort requires identifying weaknesses and gaps in knowledge. Finally, self-assessment promotes a sense of self-efficacy and control by promoting effective monitoring and use of repair strategies.

A learner model for self-assessment should incorporate both task-specific and general goal orientations. Task-specific self-assessment is clearly yoked to task goals, as discussed above for goal setting. Thus, task-specific self-assessment involves periodic judgments of progress, effort, effectiveness of employed strategies, and perceived strengths and weakness with regards to specific, proximal, and appropriately difficult goals. How best to represent these quantities is an open question. While some are easily considered as quantities on an ordinal scale (e.g., progress, effort, and effectiveness), in order for a learner model to be highly useful, it should support specific guidance and scaffolding of self-assessment. For example, if the learner notes that some strategies being employed are not effective, the system should be able to comment on the applicability of that strategy in the current context and suggest alternative strategies if appropriate. Such specific remediation is necessary to handle situations where the learner is using the correct strategy in name but is applying the strategy incorrectly. Thus, the learner model should be able to infer strategy use from learner behavior or be able to use self-report (e.g., natural language input) to identify what strategies have been used by the learner.

Self-Reflection

Self-reflection may be considered in terms of the more general goal orientations described above, such as personal interests, history of improvement, and strengths and weaknesses in a less specific and distal context. According to Zimmerman (2002), self-reflection occurs after the learning task but may involve comparisons between self-performance and performance of others or some other standard. These comparisons imply both a backward-looking judgment (at overall performance) and a forward-looking judgment (what performance remains to be achieved). Although arguably these judgments could take place during the learning task, self-reflection also includes attributions that explain overall performance and thus may be considered distinct from self-assessment (Pintrich, 2000; Zimmerman, 2002).

Attributions are causal beliefs about performance that explain performance based on factors that are either outside or within the learner's control. For example, the attribution, "I failed because I'm a stupid person" is a maladaptive attribution because it implies that change is not within the learner's control. Alternatively, adaptive attributions cast failure in terms of specific decisions, like improper strategy use. As reviewed by (Pintrich, 2000), adaptive attributions have been found to increase deeper cognitive processing and improve motivation, affect, and effort.

The general goal orientations involved in self-reflection require tracking learners over longer periods of time. Relevant variables include personal interests, history of improvement, and strengths and weaknesses in a less specific and distal context. The temporal scale of self-reflection suggests that the variables in question will change very slowly. Arguably, self-reflections at this temporal scale are more motivational and affective than cognitive. This implies that a given learner model for general self-reflection can make use of self-report and attempt to update these variables less frequently.

It is important to consider how the learner model of these variables may be used to support motivational or affective functioning. Ideally a system would be able to enhance motivation by presenting the learning task in a way that capitalizes on the learner's interests. When appropriate, the system can use the learner's own sense of personal improvement as a motivational/affective intervention, e.g., "Don't give up now – you were just saying how far you've come." In addition, learner models that track attribution may be important for preventing maladaptive attributions. For example, if the learner articulates a maladaptive attribution during self-reflection, the system would ideally challenge the maladaptive attribution and suggest an alternative explanation.

Table 4-2 summarizes the important variables for learner models.

Table 4-2. Learner model variables for self-regulated strategies

Strategy	Variables
Goal setting	Specificity Difficulty (relative to ability and domain) Proximity
Self-Assessment	Progress Effort Strategy Effectiveness Perceived Strengths and Weakness
Self-Reflection	Personal interests History of improvement Strengths and Weaknesses (Traits)

Social Constructivism

Six of the ISI strategies displayed in Table 4-1 are deeply aligned with learning in a social context. Vygotsky (1978, p. 88) championed the importance of social contexts in the development of learning, writing that "human learning presupposes a specific social nature and a process by which children grow into the intellectual life of those around them." Although just one of the many forms of social constructivism (Palincsar, 1998), Vygotsky's framework is philosophically aligned with many instructional strategies. Perhaps the single most influential idea in Vygotsky's framework is the zone of proximal development, which distinguishes between two levels of ability. The first level describes what children can do on their own. The second level, the zone of proximal development, describes what children can do with the assistance of others. Vygotsky's central claim is that abilities in the zone of proximal development will mature and move into the first level, or in his own words, "What is in the zone of proximal development today will be the actual developmental level tomorrow – that is, what a child can do with assistance today she will be able to do by herself tomorrow" (Vygotsky, 1978, p. 87). The zone of proximal development is consistent with accounts in linguistics and cognitive psychology, which emphasize that "comprehension ... must precede production" (Wood, Bruner & Ross, 1976, p. 90). It is well known that children's comprehension, or receptive capability to comprehend language, precedes their productive capability. Indeed, this is just another way of stating the zone of proximal development – children can comprehend the necessary information with the assistance of others before they can produce it on their own. Instructional strategies inspired by social constructivism and their requisite student models place great emphasis on the adaptive, collaborative, and social nature of learning.

Scaffolding and Affective ITS

The instructional strategy that is perhaps the most aligned with social constructivism is scaffolding. In scaffolding, a student's progress in a learning task is supported by a more experienced other (i.e., person, agent, system) such that scaffolding occurs in the student's zone of proximal development. Scaffolding can be seen clearly in a number of educational interventions in close concert with the instructional strategies of modeling and fading (Collins, Brown & Holum, 1991). In modeling, the instructor demonstrates the learning task, while in scaffolding, the student attempts the task with instructor support. By the time the student has reached fading, the student can engage the task with no (or extremely little) instructor support. This paradigm is also sometimes called "I do, we do, you do." If scaffolding is viewed as a blend of student actions and instructor interventions, then it is clear that it is a continuum with modeling and fading at opposite ends. Therefore, modeling and fading can be considered to be subsumed under scaffolding, representing the two extremes of scaffolded support. An early and profoundly influential study of naturalistic human tutoring describes the scaffolding process in great detail (Wood et al., 1976). According to Wood and colleagues, scaffolding is meant to draw the student into participating in the learning task to the point where the tutor can provide feedback. Over time, the tutor-provided feedback diminishes as the student achieves mastery. A high-level description of the mechanisms required to achieve scaffolding are described as follows:

> *The effective tutor must have at least two theoretical models to which he must attend. One is a theory of the task or problem and how it may be completed. The other is a theory of the performance characteristics of his tutee. Without both of these, he can neither generate feedback nor devise situations in which his feedback will be more appropriate for this tutee in this task at this point in task mastery. The actual pattern of effective instruction, then, will be both task and tutee dependent, the requirements of the tutorial being generated by the interaction of the tutor's two theories.* (Wood et al., 1976, p. 97)

Virtually all known ITSs recognize and embody these two models, generally known as the domain and learner models, which underscores the importance of scaffolding as an instructional strategy. With respect to specific variables in learner models, Wood et al. further specify the processes and mechanisms of scaffolding as consisting of six components:

1. *Recruitment*. The tutor draws the student into the task by gaining their attention, stimulating their interest, and fostering a level of commitment to the learning task.

2. *Reducing degrees of freedom.* The tutor reduces task difficulty to the appropriate level for the student's ability level.

3. *Direction maintenance*. The tutor keeps the student on task by providing affective and motivational support when needed.

4. *Marking critical features.* The tutor provides feedback, hints, and other kinds of guidance that help the student progress in the learning task.

5. *Frustration control*. The tutor prevents negative affect that would impede successful learning.

6. *Demonstration*. The tutor models the learning task based on the student's current attempt, emphasizing the task components where the student is having trouble.

The emphasis on and seamless interweaving of affective and motivational considerations into the above account of a scaffolding is noteworthy. It suggests that scaffolding divorced from these considerations has drifted away from the evidence provided by natural observations of tutoring. We therefore consider what capabilities a learner model should provide to support this entire account of scaffolding, component by component.

Recruitment requires a model of attention such that the presence or absence of a student's attention can be determined. It further requires a notion of student interest in a general sense (i.e., what do students/humans generally find interesting) and/or a notion of a particular student's interests (see *Self-Reflection* above). In either case, a model of interest informs tutor actions that make the connection between student interests and the learning task manifest to the student. Finally, recruitment involves fostering a sense of commitment to the learning task. Commitment may be relative to the student's goals, and so drawing a connection between the learning task and the student's goals provides one approach to fostering commitment (see *Goal Setting* above). However, commitment may also derive from social norms and impulses, such as a desire to please the tutor or otherwise appear competent and knowledgeable. We defer discussion of such matters to the *Pedagogical Agent* section below.

Reducing the degrees of freedom is commonly supported in learner models that assess the ability level of the student and in domain models that represent the difficulty of the material. Ideally, the learner model supports the selection of learning tasks at the appropriate level of difficulty for the student given the student's prior knowledge and progress. Direction maintenance, the third component of scaffolding, entails a model of on- and off-task behavior. Furthermore, the tutor should be able to determine if off-task behavior has resulted from a misunderstanding of the task or if off-task behavior has resulted from negative affective/motivational states leading to an overt abandonment of the learning task or a covert abandonment of the learning task, i.e., "gaming the system" (Baker, Corbett, Koedinger & Wagner, 2004). This distinction should inform whether the tutor uses a cognitive or an affective-oriented strategy to redirect the student to on-task behaviors.

The fourth component, marking critical features, is commonly implemented in learner models that have an overlay model structure (see Pavlik et al., chapter 5 in this volume). Overlay models typically associate mastery scores with elements of the domain model and update these scores based on student progress. Thus, overlay models allow the system to provide feedback to answers as well as hints that suggest what the student should consider next.

Frustration control, the fifth component, again requires a model of learner affect, both in a general sense and in an individualized sense. A general model of frustration includes such variables as affective traits, affective states, and local/global models of task difficulty. Individualized models of frustration may track whether a student has a greater or lesser tendency to become frustrated than the norm (a trait) and with the appropriate sensors even predict frustration in real time at a point in the learning session (a state). Task difficulty models of frustration may include the correlation amongst task difficulty, time on task, and frustration, taking into account that frustration may accumulate over extended periods of task difficulty. Although some analyses of intelligent tutoring systems have found that frustration plays a minor role (Baker, D'Mello, Rodrigo & Graesser, 2010), this could be attributable more to the ability of such systems to maintain appropriate task difficulty rather than to the irrelevance of frustration in learning contexts.

Finally, demonstration (modeling the learning task) requires knowing both when to demonstrate as well as how to demonstrate. Knowing how to demonstrate is a matter decided by the domain model of the learning environment. Knowing when to demonstrate, on the other hand, is an important component of the learner model that relies on tracking the student's ability as well as affective and motivational factors. If the current learning task is judged by the system to be outside the abilities of the student even when hints and feedback are given, then demonstration is warranted both to alleviate this impasse and prevent demotivation and negative affect. An overlay model that accurately assesses the difficulty of the current task, either by comparing it to tasks the student has already mastered or by considering the student's mastery of the component abilities underlying the task compared to other students, may be helpful in determining when demonstration is necessary.

Pedagogical Agent

Pedagogical agents are computer characters who play a social role in a given learning environment. As such, pedagogical agents fit well into the social constructivist perspective. Typically, pedagogical agents are tutor agents, but they may also be peer agents (see *Peer Learning* below). Given the prominence of scaffolding as an instructional strategy, tutor agents are perhaps best considered with regard to the six components of scaffolding considered above. Since humans typically treat computers as social agents and comparably to other humans (Nass, Steuer & Tauber, 1994; Reeves & Nass, 1996; Nass & Moon, 2000), tutor agents have the potential to transparently replace human tutors in computer-based learning environments like intelligent tutoring systems. But if this is the case, one might inquire whether a learning environment, by virtue of running on a computer, is by default imbued with some sort of agency in the mind of the user. If that is the case, then one may ask what pedagogical agents add over and above this default social agency.

A number of additional contributions by pedagogical agents have been described (Johnson, Rickel & Lester, 2000). Central to these contributions is the presentation of the agent as an animated agent, and sometimes as an animated agent in a virtual environment. The first contribution is the ability to present interactive demonstrations, whereby the student can watch the tutor agent demonstrate a motor skill or complex manipulation of an object. Unlike a pre-recorded movie, such a demonstration is potentially parameterizable and could be idealized according to the sixth component of scaffolding mentioned above. The second contribution of animated pedagogical agents (APAs) is the ability to use gaze and gestures to guide student attention (Lester et al., 1999). The ability to direct attention is important in the scaffolding components of recruitment, direction maintenance, and marking critical features. The third contribution, nonverbal feedback, may help with marking critical features. Finally, APAs may convey and elicit emotions, which support the scaffolding components of recruitment, direction maintenance, and frustration control. Given these considerations, it is debatable whether the addition of a pedagogical agent requires changes in the learner model that are not already accounted for by the requirements for scaffolding. If anything, the addition of pedagogical agents may have more implications for the pedagogical model because of the additional capabilities they provide for delivering instructional and affective cues.

It is important to note that APAs have the potential to support scaffolding, but only if they are designed to take advantage of these capabilities. For example, one study that did not take advantage of these capabilities as described above found essentially no differences between three versions of an intelligent tutoring system, namely, animated agent text only, versus synthesized speech only (Graesser et al., 2003). However, a similar study that used a pre-recorded human voice and gestures as the tutor worked through examples found significant learning gains compared to a text-only condition (Atkinson, 2002). Another study manipulated persona to create an emotionless "expert" agent and an emotion-displaying "mentor" agent for two versions of the same learning environment (Baylor & Kim, 2005). That study found that while the expert agent promoted learning gains and was perceived as credible, the mentor agent not only promoted learning gains but also increased self-efficacy. These various studies highlight the potential for APAs to support scaffolding.

Peer Learning

Although peer learning can take many forms, a well-known approach is "reciprocal teaching" (Palincsar & Brown, 1984; Palincsar, 1986). In this program, as instructors read the text, they think aloud to model their comprehension process to the student including their reasoning for when to use each strategy. In a classic modeling-scaffolding-fading paradigm, the instructor and student take turns as the student gradually learns the strategies and practices them while the instructor provides feedback. More specifically, students read paragraph by paragraph and generate questions, summarize, clarify terms and concepts, and make predictions about what is coming up in the text. These practices becomes a dialogue as the instructor comments on and contributes to the student's questions, summaries, and other activities, or as other students make similar contributions in small group sessions. In other words, these activities are situated in an interactive dialogue among instructor and students. In this way, reciprocal teaching creates multiple opportunities for learning: one in the role of the answering student, one in the role of an observing student, and one in the role of the teacher. Each of these roles has a slightly different implication for learner models.

Like tutor agents, peer pedagogical agents may assume social roles in a given learning environment (Kim & Baylor, 2006b), including teachable agent and peer agent. Teachable agents typically have a knowledge representation that human students can directly manipulate (Reif & Scott, 1999; Biswas, Schwartz, Leelawong & Vye, 2005). A well-known teachable agent system is Betty's Brain (Biswas et al., 2005; Leelawong & Biswas, 2008), in which students directly manipulate a concept map (the brain) in order to teach Betty. In Betty's Brain, the domain modeled is typically a causal system in nature, such as ecosystems, the circulatory system, or climate change. Human students use an e-textbook to learn the information to teach Betty. Once a student teaches Betty by building a concept map, Betty can employ qualitative reasoning to reason through a chain of links. This allows Betty to answer questions, take a quiz, or explain her reasoning. Recent experiments indicate that students who teach Betty spend more time on learning tasks compared to students working with the same system but without Betty (Chase, 2011), suggesting that a teachable agent can increase student motivation to learn. The learner model for Betty's Brain mostly centers around the concept map, which is a kind of overlay model on an unseen "expert map," and so learner modeling in this scenario is largely congruent with the typical case in intelligent tutoring systems. However, the special role of human student as teacher can create new opportunities for learner modeling, including teaching-based activities like using the e-book, building the concept map, and taking quizzes. Kinnebrew et al. (2013) explore patterns of student interaction and their implications for student learning. Since the human student can decide what task to perform at any time, they may exhibit learning-promoting behaviors, like careful reading, concept map building, and quiz-taking, or they may exhibit maladaptive behaviors like rapidly alternating between skimming the text, incremental concept map building, and many cycles of quiz failing and re-taking. In systems like Betty's Brain, it becomes even more important to track user interactions to infer what self-regulation strategies (if any) the student is using. These variables are complementary to the self-regulation variables described in Table 4-2.

Peer agents, unlike teachable agents, do not actually learn but instead model behavior to the human student. Peer agents can give answers, express misconceptions, or display affect (Millis et al., 2011). Although multiple dimensions describing peer agents have been proposed (Kim & Baylor, 2006b), perhaps the two most important dimensions are competence and responsiveness. Competency refers to the correctness or quality of the information provided by the peer agent. Responsiveness refers to the initiative the peer agent demonstrates, typically whether it proactively gives advice or not. Note that these dimensions may be applied symmetrically to the human student who is a peer of the agent. A study examining competency and responsiveness found differential effects according to learning outcomes and self-efficacy (Kim & Baylor, 2006a). Students with competent peer agents learned more, but students with less competent peer agents expressed greater self-efficacy. Students with proactive peer agents scored higher on tests of recall than students with low responsive peer agents. While overall these results may suggest that peer agents should be competent and proactive, the study's authors argue that the human student's ability level should be considered when setting the competency or responsiveness of a peer agent. One system combines this suggestion with teachable agents to create a system that enters one of three modes (regular ITS, teachable agent, and vicarious learning) depending on the current ability level of the student (Millis et al., 2011). Significant differences in learning gains in delayed post-test have been found between vicarious and non-vicarious conditions, and it is hypothesized that high ability students should benefit most from teaching agents, and low ability students should benefit most from vicariously observing peer agents model correct behavior.

Table 4-3 summaries important learner variables for the instructional strategies described in this section. Parentheses indicate variables that overlap with previously stated variables in Table 4-2 or that are typically considered part of the domain model.

Table 4-3. Learner model variables for social constructivist strategies

Strategy	Variables
Recruitment	Attention General interest Specific student interest (self-reflection) Commitment (goal setting)
Reducing Degrees of Freedom	Student ability Task difficulty (domain model)
Direction Maintenance	On/off task behavior
Marking Critical Features	Domain model
Frustration Control	Affective traits Affective states Session difficulty (domain model) Task difficulty (domain model)
Demonstration	Task difficulty (domain model) Student ability
Teachable agents	Patterns of interaction to infer strategies
Peer agents	Student ability

Improving Memory

Memory is one of the longest studied topics in the history of psychology (Ebbinghaus, 1913). The range of topics in memory research is immense, ranging from language-based effects to multimedia effects (Byrne, 2008). For the purposes of the present discussion, two topics are particularly relevant, namely, how the temporal structure of practice affects recall and how multimedia presentation affects recall.

Distributed Practice and Massed Practice

The temporal structure of practice is primarily concerned with (1) the duration of study for an item, (2) the temporal interval, or spacing, between repeated presentations of an item, and (3) the retention interval, or time between the last study episode and a retention test (Cepeda, Pashler, Vul, Wixted & Rohrer, 2006). Given these distinctions, massed practice describes practice that emphasizes duration of study of a single item, whereas distributed practice describes practice that emphasizes the spacing between repeated presentations of an item. A recent meta-analysis reports two main finds with respect to distributed and massed practice (Cepeda et al., 2006). First, distributed practice resulted in more correct responses than massed practice for retention intervals ranging from less than 1 min to over a month when study time was held constant. It should be noted that this comparison to “pure” massed practice is unlikely to be consistent with realistic cramming before an exam, which would involve a single session of distributed practice for particular items. Second, for distributed practice, increasing the temporal interval between practices increases accuracy at retention up to a point, but then decreases accuracy after that point, and this optimum spacing increases as the retention interval increases. In other words, for a given retention interval, there is an optimum spacing of practice that maximizes recall. For longer retention intervals, the optimum spacing is longer, and the optimum spacing is shorter for shorter retention intervals. These temporal effects suggest that it is important for learner models to track the presentation history of individual items in order to maximize recall.

Moreover, available research also suggests that individual items can be optimally spaced to maximize the efficiency of learning per unit time (Pavlik & Anderson, 2005, 2008). The basic version of this model, which exhibits good fits to five different human experiments, is based on the declarative memory model of Adaptive Control of Thought-Rational (ACT-R) (Anderson & Lebiere, 1998). Unlike the ACT-R model, which assumes that forgetting is constant, Pavlik and Anderson’s basic model assumes that the forgetting rate is a function of the activation of an item. An elaborated version of this basic model considers how to best balance spacing with studying, such that recall is optimized for a given period of study (Pavlik & Anderson, 2008), which we summarize below. This optimization is important because spacing and studying are somewhat at odds. Longer spacing improves long-term recall on retention tests, but short-term is likely to increase forgetting and, by implication, restudying. However, restudying takes time, so a successful recall trial takes less time than an unsuccessful trial that necessitates restudying.

Using the extended ACT-R model, Pavlik and Anderson define a learning rate measure by which they can select the optimal item for practice on any given trial. The learning rate is based on the predicted activation gain for an item at the retention interval, but normalized for the time cost to practice that item. At any given moment, items will have different learning rate values, but one will be closest to its maximum learning rate. Although the items are not explicitly compared with each other to select the optimal item, because the basic model limits the gains of activation for items that are already highly activated, the model has a preference for items with relatively low activation. In experimental comparisons, the optimized schedule of practice was significantly better in terms of recall and latency, with large effect sizes, than a flashcard-based system and another optimized schedule that predicted recall as a function of practice. These findings suggest that both the presentation history of items (correct trials and incorrect+restudy trials for each item) and the time cost to practice any item should be considered to optimize recall. Specific learner model variables include item probability of correctness, response latency, and failure latency.

Multimedia Instruction

Multimedia consists of auditory and visual representations of words and images. Examples of multimedia include illustrated books, television, and computer-based learning environments. According to Mayer (2008), the fundamentals of multimedia learning are that (1) we have dual channels for processing auditory and visual stimuli, (2) we have limited capacity to process information at a given point in time, and (3) we learn deeply by organizing and integrating information during learning. Mayer elaborates these fundamentals in ten principles that together optimize the active processing of information by respecting the limitations of cognitive processing.

We briefly summarize these ten principles, each of which specifies how information should be presented to the learner:

1. Extraneous material should be reduced to the extent possible
2. Essential material should be highlighted by overviewing the main ideas
3. Redundancy in separate channels helps learning, but redundancy in the same channel hurts learning.
4. Spatial contiguity should be preserved, e.g., by placing words next to their respective images.
5. Temporal contiguity should be preserved by synchronizing presentation of related information that is in different channels.
6. Material should be presented in small chunks.
7. Pretraining should be used to basic vocabulary and concepts before attempting to teach a complex system.
8. Spoken text and images should be used instead of printed text and images.
9. Spoken text and pictures should be used instead of either alone.
10. Material should be presented in a personalized conversational style.

The extent to which the principles of multimedia instruction align with learner models is somewhat suspect. These principles state general constraints with regard to human cognition, and so do not represent variables that would be tracked for an individual in order to optimize instruction for that individual. The theory of learning styles has been proposed, by which individuals may vary in how they best learn from different modes of information presentation, but little evidence has been found to support these claims (Pashler et al., 2008). Therefore, multimedia principles may be best applied to the domain or pedagogical models – in the design of the instruction itself.

Table 4-4 summarizes the important learner model variables discussed in this section.

Table 4-4. Learner model variables for improving memory

Strategy	Variables
Practice	Number of trials Delay between trials
Optimal practice	Probability of correctness Response latency Failure latency

Conclusion

In this chapter, we examined the top ten strategies of the ISI from the perspective of learner models. Most of the ten strategies could be associated with concrete variables that a learner model must include in order to support that strategy. This was true for the self-regulated strategies that included variables for proximal and distal goals, personal interests, history of improvement, attributions, and strategy use either through self-report or inference from student behavior. The social-constructivist strategies include the classic learner model variables associated with scaffolding such as overlay models on the domain that specify the difficulty level of material and the student's ability, student interests, affect, goals, and on-off task behavior. Pedagogical agents, while providing potential affective and gestural support for scaffolding, have little correspondence to the learner model except perhaps by virtue of learner preferences. Peer learning has different implications for learner models depending on whether the peer is a teachable agent (with the human student as the teacher) or a collaborative peer. Teachable agents create a context for a host of new variables tracking the strategies and behaviors used by the human student (e.g., quizzing the teachable agent or studying an e-book in preparation of teaching the agent). Use of collaborative peer agents may benefit from learner models that track the ability level and self-efficacy of the human student, but this question awaits future research. Memory-enhancing presentation strategies likewise differ on their learner modeling needs. Distributed practice (which subsumes massed practice in real-world contexts) at the extreme can require extensive item level variables that track the accumulation of practice and the rate of forgetting. Multimedia instruction, in contrast, is guided by a set of principles that are not learner-specific. As a result they are perhaps best situated outside the learner model.

Author Note

This research was supported by the National Science Foundation (NSF) (BCS 0826825 and DRL 1108845). Any opinions, findings and conclusions, or recommendations expressed in this paper are those of the author and do not represent the views of the NSF.

References

Anderson, J. R. & Lebiere, C. J. (1998). The atomic components of thought. Lawrence Erlbaum.

Atkinson, R. K. (2002). Optimizing learning from examples using animated pedagogical agents. Journal of Educational Psychology, 94 (2), 416.

Baker, R. S. J. d., Corbett, A. T., Koedinger, K. R. & Wagner, A. Z. (2004). Off-task behavior in the cognitive tutor classroom: when students "game the system". In Proceedings of the sigchi conference on human factors in computing systems (pp. 383–390). New York, NY, USA: ACM. Available from http://doi.acm.org/10.1145/985692.985741

Baker, R. S. J. d., D'Mello, S. K., Rodrigo, M. T. & Graesser, A. C. (2010). Better to be frustrated than bored: The incidence, persistence, and impact of learners' cognitive-affective states during interactions with three different computer-based learning environments. International Journal of Human-Computer Studies, 68 (4), 223–241. Available from http://dx.doi.org/10.1016/j.ijhcs.2009.12.003

Baylor, A. L. & Kim, Y. (2005, April). Simulating instructional roles through pedagogical agents. International Journal of Artificial Intelligence and Education, 15 , 95–115. Available from http://dl.acm.org/citation.cfm?id=1434925.1434927

Biswas, G., Schwartz, D., Leelawong, K. & Vye, N. (2005, March). Learning by teaching: A new agent paradigm for educational software. Applied Artificial Intelligence, 19 , 363–392.

Byrne, J. H. (2008). Learning and memory: a comprehensive reference. Elsevier. Available from http://books.google.com/books?id=bzUQAQAAIAAJ

Cepeda, N. J., Pashler, H., Vul, E., Wixted, J. T. & Rohrer, D. (2006). Distributed practice in verbal recall tasks: A review and quantitative synthesis. Psychological bulletin, 132 (3), 354–380.

Chase, C. C. (2011). Motivating persistence in the face of failure: The impact of an ego-protective buffer on learning choices and outcomes in a computer-based educational game. Unpublished doctoral dissertation. Available from http://books.google.com/books?id=uHsdh6mKHuYC

Cohen, J. (1992). A power primer. Psychological bulletin, 112 (1), 155–159.

Collins, A., Brown, J. S. & Holum, A. (1991). Cognitive apprenticeship: Making thinking visible. American Educator, 15 (3), 38–46.

Ebbinghaus, H. (1913). Memory: A contribution to experimental psychology. Teachers College, Columbia University. Available from http://books.google.com/books?id=oRSMDF6y3l8C

Graesser, A. C., Moreno, K., Marineau, J., Adcock, A., Olney, A. & Person, N. (2003). AutoTutor improves deep learning of computer literacy: Is it the dialog or the talking head? In U. Hoppe, F. Verdejo & J. Kay (Eds.), Proceedings of artificial intelligence in education (p. 47-54). Amsterdam: IOS Press.

Johnson, W., Rickel, J. & Lester, J. (2000). Animated pedagogical agents: Face-to-face interaction in interactive learning environments. International Journal of Artificial Intelligence in Education, 11 , 47–78.

Kim, Y. & Baylor, A. L. (2006a). Pedagogical agents as learning companions: The role of agent competency and type of interaction. Educational Technology Research and Development, 54 (3), 223–243.

Kim, Y. & Baylor, A. L. (2006b). A social-cognitive framework for pedagogical agents as learning companions. Educational Technology Research and Development, 54 (6), 569–596.

Kinnebrew, J.S., Loretz, K.M. & Biswas, G. (2013). A Contextualized, Differential Sequence Mining Method to Derive Students' Learning Behavior Patterns. Journal of Educational Data Mining.

Leelawong, K. & Biswas, G. (2008). Designing learning by teaching agents: The Betty's Brain system. Int. J. Artif. Intell. Ed., 18 (3), 181–208.

Lester, J. C., Voerman, J. L., Towns, S. G. & Callaway, C. B. (1999). Deictic believability: Coordinating gesture, locomotion, and speech in lifelike pedagogical agents. Applied Artificial Intelligence, 13, 383–414.

Locke, E. A. & Latham, G. P. (2002). Building a practically useful theory of goal setting and task motivation: A 35-year odyssey. American Psychologist, 57 (9), 705 -717. Available from http://ezproxy.memphis.edu/login?url=http://search.ebscohost.com/login.aspx?direct=true&db=pdh&AN=2002-15790-003&site=ehost-live

Mayer, R. E. (2008). Applying the science of learning: evidence-based principles for the design of multimedia instruction. American Psychologist, 63 (8), 760–769.

Millis, K., Forsyth, C., Butler, H., Wallace, P., Graesser, A. & Halpern, D. (2011). Operation aries!: A serious game for teaching scientific inquiry. In M. Ma, A. Oikonomou & L. C. Jain (Eds.), Serious games and edutainment applications (pp. 169–195). Springer.

Nass, C. & Moon, Y., 2000. Machines and mindlessness: Social responses to computers. Journal of Social Issues, 56(1), 81–103.

Nass, C., Steuer, J. & Tauber, E. R. (1994). Computers are social actors. In Proceedings of the SIGCHI conference on human factors in computing systems: celebrating interdependence (pp. 72–78). New York, NY: ACM. Available from http://doi.acm.org/10.1145/191666.191703

Palincsar, A. S. (1986). The role of dialogue in providing scaffolded instruction. Educational Psychologist, 21 (1-2), 73–98.

Palincsar, A. S. (1998). Social constructivist perspectives on teaching and learning. Annual review of psychology, 49 (1), 345–375.

Palincsar, A. S. & Brown, A. L. (1984). Reciprocal teaching of comprehension-fostering and comprehension-monitoring activities. Cognition and Instruction, 1 , 117-175.

Paris, S. G. & Paris, A. H. (2001). Classroom applications of research on self-regulated learning. Educational psychologist, 36 (2), 89–101.

Paris, S. G. & Winograd, P. (1999). The role of self-regulated learning in contextual teaching: Principles and practices for teacher preparation (Vol. 16; Information Series No. 376). Washington, DC: ERIC Clearinghouse on Adult, Career, and Vocational Education.

Pashler, H., McDaniel, M., Rohrer, D. & Bjork, R. (2008). Learning styles concepts and evidence. *Psychological science in the public interest*, *9*(3), 105-119.

Pavlik, P. I. & Anderson, J. R. (2005). Practice and forgetting effects on vocabulary memory: An activation-based model of the spacing effect. Cognitive Science, 29 (4), 559–586.

Pavlik, P. I. & Anderson, J. R. (2008). Using a model to compute the optimal schedule of practice. Journal of Experimental Psychology: Applied, 14 (2), 101–117.

Pintrich, P. R. (2000). The role of goal orientation in self-regulated learning. In M. Boekaerts, P. R. Pintrich & M. Zeidner (Eds.), Handbook of self-regulation. Burlington, MA: Elsevier.

Reeves, B. & Nass, C. I. (1996). The media equation: How people treat computers, television, and new media like real people and places. New York, NY: Cambridge University Press.

Reif, F. & Scott, L. A. (1999). Teaching scientific thinking skills: Students and computers coaching each other. American Journal of Physics, 67 (9), 819–831.

Schunk, D. H. (1990). Goal setting and self-efficacy during self-regulated learning. Educational psychologist, 25 (1), 71–86.

Tarr, Ron. (2012). Instructional Strategies Indicator. Retrieved May 12, 2010 from http://instructional-technology.info.

Vogel-Walcutt, J., Fiorella, L. & Malone, N. (2012). Instructional strategies framework for training systems: A review and analysis of the literature. http://instructional-technology.info/review/main search/ IS-paper-4-18-11initialFinal.pdf. (Accessed: 11/15/2012)

Vogel-Walcutt, J., Malone, N. & Fiorella, L. (2012). Instruction specific model for strategy selection. http://instructional-technology.info/review/ main search/ISI FrameworkPaper Draft.pdf. (Accessed: 11/15/2012)

Vygotsky, L. S. (1978). Mind in society. Cambridge, MA: Harvard University Press.

Wood, D., Bruner, J. & Ross, G. (1976). The role of tutoring in problem solving. Journal of child psychology and psychiatry, 17 (2), 89–100.

Zimmerman, B. J. (2002). Becoming a self-regulated learner: An overview. Theory into practice, 41 (2), 64–70.

CHAPTER 5 –A Review of Student Models Used in Intelligent Tutoring Systems

Philip I. Pavlik Jr.[1], Keith Brawner[2], Andrew Olney[1], and Antonija Mitrovic[3]
[1]University of Memphis, [2]U.S. Army Research Laboratory, [3]University of Canterbury - New Zealand

Introduction

Generally, an ITS is considered to have four major components (Elsom-Cook, 1993; Graesser, Conley & Olney, 2012; Nkambou, Mizoguchi & Bourdeau, 2010; Psotka, Massey & Mutter, 1988; Sleeman & Brown, 1982; VanLehn, 2006; Woolf, 2008): the domain model, the student model, the tutoring model, and the tutor-student interface model.

- The *domain model* contains the set of skills, knowledge, and strategies of the topic being tutored. It normally contains the ideal expert knowledge and may also contain the bugs, mal-rules, and misconceptions that students periodically exhibit. It is a representation of all the *possible* student states in the domain. While these states are typically tied to content, general psychological states (e.g., boredom, persistence) may also be included, since such states are relevant for a full understanding of possible pedagogy within the domain.

- The *student model* consists of the cognitive, affective, motivational, and other psychological states that are inferred from performance data during the course of learning. Typically, these states are summary information about the student that will subsequently be used for pedagogical decision making. The student model is often viewed as a subset of the domain model, which changes over the course of tutoring. For example, “knowledge tracing” tracks the student’s progress from problem to problem and builds a profile of strengths and weaknesses relative to the domain model (Anderson, Corbett, Koedinger & Pelletier, 1995). Since ITS domain models may track general psychological states, student models may also represent these general states of the student.

- The *pedagogical model* takes the domain and student models as input and selects tutoring strategies, steps, and actions on what the tutor should do next in the exchange with the student to move the student state to more optimal states in the domain. In mixed-initiative systems, the students may also initiate actions, ask questions, or request help (Aleven, McLaren, Roll & Koedinger, 2006; Rus & Graesser, 2009), but the ITS always needs to be ready to decide “what to do next” at any point and this is determined by a tutoring model that captures the researchers’ pedagogical theories. Sometimes what to do next implies waiting for the student to respond.

- The *tutor-student interface model* interprets the student’s contributions through various input media (speech, typing, clicking) and produces output in different media (text, diagrams, animations, agents). In addition to the conventional human-computer interface features, some recent systems have had natural language interaction (Graesser, D’Mello, et al., 2012; Johnson & Valente, 2008), speech recognition (D’Mello, Graesser & King, 2010; Litman, 2013), and the sensing of student emotions (Baker, D’Mello, Rodrigo & Graesser, 2010; D’Mello & Graesser, 2010; Goldberg, Sottilare, Brawner & Holden, 2011).

This review focuses on the approaches to student modeling, with particular attention to the implications of these models for design of GIFT. GIFT similarly adopts this four-part distinction, but refers to the models as modules, since GIFT is an actual software framework that reifies each of these models with modules in the software. GIFT also includes an optional sensor module, as a way to measure user states during interactions with the tutoring system.

This chapter specifically focuses on the student model aspect of the ITS. In order to give a clear perspective on the student model, it is essential to also address the domain, pedagogical, and interface models because the models do not function in isolation. The connections between domain and student models are tightly coupled. Less tightly coupled, but still substantial, is the dependence of the pedagogical model on the student model. In contrast, the tutor-student interface model, while it clearly relates to the domain and pedagogical models, is beyond the scope of this article. The structure of the domain model may be necessary to consider in the context of student modeling, and to some small degree, the pedagogical model. However, this chapter does not describe the complications of mapping domains to interfaces or describing how different interfaces reify similar pedagogy. We leave interface issues to subsequent volumes of this book series.

Assessing a Student Model - Criteria

When assessing models, it is appropriate to emphasize the limitations of complexity and fidelity of representation. Consequently, all models of learning are wrong in certain ways because they cannot hope to capture the full complexity of the real student's mind. In the GIFT system, we want the designer to implement the most useful student models, not necessarily the most correct student models. For example, we might suppose that a multilevel neural network, such as the Leabra architecture (O'Reilly, 1998), would be a more correct representation of a student. However, it would not be very useful because it would be monumentally complex and impractical to use it for clear pedagogical inference and tracking of the domain model. We therefore evaluate the student model options in regard to several criteria of usefulness. These criteria include the following:

- *Student model fit to data.* This is a simple validity criterion that refers to how well the student model can be used to simulate the quantitative and/or qualitative patterns of learning in real students. Issues of model fit have been reviewed by Desmarais and Baker (2012) and more generally by Schunn and Wallach (2005).

- *Ease of understanding*. The reality behind educational system design is that student modeling methods need to be comprehensible to system designers and builders. Therefore, student modeling techniques that require optimizing a Bayesian network using simulated annealing may have limited applicability unless such complexity is easily approached by system designers. This limitation of complex models parallels the tendency for complexity of student models in running systems to lag behind the complexity of student modeling techniques presented in research. This explains why existing systems are often less complex from systems invented 60 years ago (Smallwood, 1962).

- *Generality/flexibility*. Many of the student models we review have only one or two primary successful contexts, so some student models have limited generality. Generality may have a downside in that more general models tend to be simpler, and therefore, lose power in specific domains as they gain generality. To what extent can student models be reused in new contexts to improve the adaptation to students? To what extent is a student model essentially chained to specific domain content, so that after this domain content is learned, the student model is no longer relevant?

- *Cost of creation*. Research often cites the high costs of creating content (Aleven, McLaren, Sewall & Koedinger, 2006), and such costs also apply to student models. If a student model requires 2 years of data collection to determine the parameters, the cost may be prohibitive. The issue of cost refers to person hours of production, while the issue of ease of construction involves how skilled (professional degree) those workers need to be.

- *Granularity*. Grain-size may refer to steps, single problems, single curriculum units, multiple curriculum units, whole domains, or multiple domains. Student models are typically applied to pedagogical decisions at one or more levels of granularity. For example, a student model might trace the state of the student within the steps of a problem, or how that state changes across problems, units, or even different domains. One hypothesis we make in this chapter is that different models may be more or less useful depending on the granularity of the pedagogical decisions.

- *Time scale*. Time scale refers to the overall longevity of the student model.. A model of student state at the step level might be created from only the current problem interaction, or it may take into account prior actions in prior problems or earlier units. The time scale helps determine whether an ITS system is actually a collection of independent units, or whether the student model is cumulative across the domain as the student progresses.

- *Learning gains in practice*. Evidence of learning gains that are directly caused by pedagogical inferences from the student model states is obviously an excellent feature for a student model. However, it is very easy to argue that learning gains depend on far more than the student model, since the pedagogical model and tutor-student interface feedback need to be appropriate for learning gains. Furthermore, comparing learning gains carefully across different control conditions, domains, research groups, and populations is difficult if not impossible to do in a valid way.

To make the assessment according to these criteria requires us to group the student models into a manageable number of categories whose members share deep similarities. One can make finer distinctions in the categories we review, but these finer distinctions may not move us toward our goal of making recommendations for GIFT. Finer distinctions do not highlight the overarching software architectural issues involved in providing support for broad range of very different student models.

Programmed Student Models

Our first category has a long history that traces back first to Pressey in 1926, who is credited with the first "teaching machine," which was little more than a question and answer device with multiple choice scoring and immediate feedback (Lumsdaine & Glaser, 1960). This initial start in the ‘‘teaching machine" movement gained little attention until the 1950 when an explosion of automated teaching theories began to emerge from various sources. Most notable was work by Skinner looking at linear programming of instruction (not to be confused with the unrelated mathematical optimization procedure) and Crowder's work with branching programs (Thomas, Davies, Openshaw & Bird, 1963).

Skinner's contribution came from his theory of behaviorism and the concept of error-free learning. While a teaching machine with a linear program does provide feedback, the key to linear programming was to provide a sequence of tasks where the sequence faultlessly provides the prerequisites in a serial order such that a prepared student might be expected to proceed through the content without errors. To accomplish this goal, detailed sequences of rule and exemplar practices are composed by the linear program designer. These sequences can be derived from a task analysis that uses a rule by rule matrix to determine associated rules as well as rules that need to be discriminated. From this rule matrix, the linear programmer then lays out the faultless sequence, which is composed of several types of cloze (fill in the blank items) including examples, rules, generalizations, and discriminations in the domain (Lumsdaine & Glaser, 1960; Thomas et al., 1963). Frequent reinforcement of prior learning was built into such sequences.

One clear issue with linear programming is that it mandates that a single sequence is optimal for all students. Crowder challenged this assumption with similar systems that branched extensively based on the student responses, thus providing a basic means of individualizing instruction. Based upon student actions, a student may be "branched" to a review, remediation, or additional content (Crowder, 1959; Lumsdaine & Glaser, 1960; Thomas et al., 1963).

This category of student models as part of the discussion of ITSs raises the question of what we should be counting as a student model. In other words, how smart does an "intelligent" tutoring system need to be to qualify as a true ITS. While this question may have philosophical interest in that it deals with the general question of what is intelligence, it does not seem to be a pragmatically helpful criterion. More sensibly, we might ask how adaptive a system is by characterizing how much freedom the domain model allows in measuring student states to infer one of many pedagogical choices. Typically, branched programming student models offer less adaptivity because student states are hardcoded as triggers for specific pedagogies since the development process for branched models handcrafts each state branch in the student model space. Perhaps these systems do not qualify as ITSs, considering the student state space is small and that pedagogical options are few.

Prior efforts such as this are particularly relevant since they help us see that new methods, such as knowledge space theory (discussed further in the chapter), can be traced to antecedent methods of much older pedigree. Indeed, sometimes these methods come back in similar form to their prior versions. For example, work with "example tracing tutors" at the Pittsburgh Science of Learning Center (PSLC) seems to involve a very advanced method for creating branching tutors by mapping out the possible "example" paths a student may take in solving a problem. They argue that this work should be considered an ITS because "example-tracing tutors are capable of sophisticated tutoring behaviors; they provide step-by-step guidance on complex problems while recognizing multiple student strategies and (where needed) maintaining multiple interpretations of student behavior." (Aleven, McLaren, Sewall & Koedinger, 2009)

The main strength of programming methods is their simplicity of execution. Linear programming is the perhaps the most simple, but arguably so simple that it misses the benefit of the many important pedagogical moves that depend on attending to individual differences. Branched programming address this shortcoming and is made easier when supported by technologies that allow branched program authoring (Aleven et al., 2009). Where branched programming begins to break down is when branching rules for addressing different student model states with different pedagogies become very sophisticated. In this case, the fit of the branched programming model to student states becomes difficult to measure, since it is idiosyncratic. Subsequent generality will then usually be very low. So, if a general system is desired it makes sense to implement adaptation (pedagogy) using methods other than hard coding the student states and their resulting pedagogy as branches in the domain model. We discuss many of these alternative means for adaptation in the next sections.

Student models that consist of branching states now have a more than 50-year history. Despite their shortcomings, they should be included in GIFT. Branched programming provides considerable power and efficiency to a developer with a simple project, perhaps explaining why SCORM includes some simple criterion based branching capability, with new proposals in the SCORM domain advocating deeper personalization through more complex rule based branching (Rey-López, Fernández-Vilas, Díaz-Redondo, Pazos-Arias & Bermejo-Muõz, 2006). However, the main advantage for robust support of branching in GIFT may be in the potential ease of construction aspect. If GIFT provides a powerful and easy mechanism to create adaptive branching tutors (perhaps by allowing the designer to graphically author possible student moves in the state space), there is the potential to attract a wider user base, with consequent wider impact on the needs of the education and training community.

Overlay Student Models

The origin of overlay models is shrouded in history that reaches back to the heyday of behavioristic thought in the late 50s and earlier. During these early years of educational software, designers were still heavily influenced by the idea that speculating on "skills" and other unseen constructs was bad science, as had been taught by Skinner. Fortunately however, some were beginning to speculate on deeper adaptivity in "teaching machines." In particular, Smallwood (1962) should probably receive credit for his work in setting the stage for concept overlay models, Bayesian knowledge tracing (BKT), and economic optimization of practice. To illustrate the groundbreaking nature of Smallwood (1962), it is useful to quote a few points from the monograph where he proposes this structure (parenthetical comments added):

1. *The decomposition of the subject matter into a set of concepts that the educator would like to teach to the student.* (Part of the domain model.)

2. *A set of test questions, for each concept, that adequately tests the student's understanding of the concept.* (Part of the domain model.)

3. *An array of information blocks, for each concept that can be presented to the student in some order (to be decided by the teaching machine)- and thus provide a course of instruction to the student on the concept.* (Part of the domain model.)

4. *A model that can be used to estimate the probability that a given student with a particular past history will respond to a given block or test question with a particular answer*. (The student model.)

5. *A decision criterion upon which to base the decisions mentioned in 3.* (The pedagogical model.)

While Smallwood continued his explorations of optimal instructional policies (1971), Richard Atkinson, a Stanford professor of psychology, who would later be nominated to head the NSF from 1975–1980 and later head the University of California System, was busy over the 1960s testing a variety of Markov models that captured learning in probabilistic state transition networks (Atkinson & Crothers, 1964; Calfee & Atkinson, 1965; Groen & Atkinson, 1966). Atkinson endorsed many of Smallwood's ideas, but perhaps described them in a more accessible form in his "theory of instruction," which he hoped to formalize in computerized instructional systems (Atkinson, 1972a). Furthering these goals, he produced an excellent demonstration of the utility of a single skill per item Bayesian overlay model, which he used to optimize the learning of German vocabulary items. The strong results supported a student model, which individualized instruction by using different parameters for each vocabulary item to control practice sequencing, obtaining 79% performance at a 1-week test, compared to 38% performance for a random sequence of practice (Atkinson, 1972b).

Work such as Atkinson's, where there is a relatively complex mathematical model that tracks a collection of independent item's correctness (the simplest overlay), has been further exemplified by work with the optimization of Japanese-English language pairs (Pavlik Jr. & Anderson, 2008) by taking account of time costs in the model so as to make pedagogical decisions not just in reference to learning, but also in reference to the time costs of this learning. Pavlik's technology has been implemented in the X-Germs series of arithmetic games from K-12, Inc. This work is arguably unique because of the complexity of the mathematical model that captures several principles of memory and treats all the learned units as independent declarative chunks. The mathematical formalism is based on ACT-R's declarative memory system and involves using the full history of practice to compute when each item is at the temporal sweet spot of optimal learning. Optimality is achieved by maximizing the expected long-term learning gains against the current cost of additional practice.

Even the early work on overlay models makes clear that an overlay student model tracks some pedagogically relevant quantity as a function of the overlay type (in this case, skills, facts, or concepts). Modern work continues to follow this pattern despite at least three different kinds of overlay models that went beyond this early work with learning sets of independent items. These methods include rule space methods, model tracing, and constraint-based models. To add further complexity, however, each of these methods has a variety of ways in which knowledge strength or probability is traced. To simplify the issue, we associate each of the three methods with its most common long-term model tracing formalisms (e.g., BKT), but keep in mind that the overlay model is loosely coupled to the equation that tracks the strength of overlaid model states. In other words, BKT could be used in a rule space context, just as logistic regression could be used to trace the strength of model traced productions.

Rule Space Student Models

An important early student model was the rule space method, which was introduced as a modification of Item Response Theory (IRT) to provide a way to categorize misconceptions in signed number arithmetic problem solving data (Tatsuoka, 1983). This original approach has stimulated successively more advanced efforts to develop the IRT approach as a student modeling method. Perhaps most important of these efforts was the development of the idea that the count of prior learning attempts can be used to predict later performance. This model has been developed by various authors and is often referred to as the additive factors model (AFM; Draney, Pirolli & Wilson, 1995; Spada & McGraw, 1985).

More recently, researchers have come to refer to the rule space mapping as a Q-matrix, which stands for "question matrix" to capture how it maps each of the questions to a number of proficiencies or misconceptions (Barnes, 2005; Barnes, Stamper & Madhyastha, 2006; Pavlik Jr., Yudelson & Koedinger, 2011). Typically, a Q-matrix is a binary matrix that maps specific skills/ rules to particular questions or types of questions. A main point of this section on overlay models is that the Q-matrix representation could be considered a standard way to represent the domain model semantics. Once a Q-matrix has been determined for some content domain, the main question becomes what mathematical formalism (typically, Bayesian-Markov process models or logistic regression) should be used to represent the student states relative to the possibilities the Q-matrix admits. The point of determining the Q-matrix with parameterized equations is that this allows the ITS to mathematically infer an order of problems that allows learning to be meticulously adapted based on prior knowledge with the goal of diagnosing and providing instruction on the "rules" of the domain, as represented in the matrix.

Q-matrix models with logistic regression quantitative tracking have become part of the modeling apparatus the PSLC (DataShop) though they have not been employed in any running systems we are aware of because the model is not adaptive (AFM, Draney et al., 1995; Spada & McGraw, 1985). However, recent work to improve the fit of the additive logistic regression models has shown the importance of capturing success and failure in this type of model, demonstrating that logistic regression can be at least as accurate as the standard BKT (Corbett & Anderson, 1992; Gong, Beck & Heffernan, 2010; Pavlik Jr., Cen & Koedinger, 2009). In this performance factors analysis (PFA) model (an AFM logistic regression variant), there are two fundamental categories of prior practice, namely, success and failure (Pavlik Jr. et al., 2009), which contribute differently to future performance, unlike in the AFM model where merely the count of prior experience is tracked. The better fit of the PFA model is sensible, since the psychological literature shows successes (in contrast to review after failing) may lead to more production-based learning and/or less forgetting (Carrier & Pashler, 1992; Karpicke & Roediger, 2008; Pavlik Jr., 2007; Thompson, Wenger & Bartling, 1978) and may lead to automaticity for shallow learning tasks (Peterson, 1965; Rickard, 1997, 1999; Segalowitz & Segalowitz, 1993). Moreover, the logic of representing additional factors of learning easily in the logistic regression model is resulting in further model variants that capture a variety of instructional differences between different student interactions (M. Chi, Koedinger, Gordon, Jordan & VanLehn, 2011) and transfer between related items (Pavlik Jr. et al., 2011). These models may have the advantage of incorporating multi-factor complexity more easily than BKT models. In general, this analytic approach attempts to decompose practice contexts to understand what specific features of the practice can be useful for predicting future performance (Beck & Mostow, 2008). Typically, such work is theory driven.

Model Tracing Student Models

Model tracing comes from Anderson's work on the LISP programming tutor at Carnegie Mellon University (CMU), which used what he called a variant of Atkinson's (1972b) work (Anderson, Conrad & Corbett, 1989). This approach created a student model that both quantitatively traced the learning of production rules in a LISP programming language tutoring system at the same time as it traced the students' responses. Production rules are low-level cognitive steps in a problem solution encoded as IF-THEN rules. For example:

IF the goal is to prove ΔABC I congruent to ΔDBE and AE and CD are collinear

THEN infer ∠ABC is congruent to ∠DBE because they are vertical angles

Reactivity to user responses was engineered by a mechanism called model tracing. Model tracing has developed from several implications of the ACT theory. These principles include (1) encoding the student model as a set of production rules, (2) communicating these rules through problem-solving exercises, (3) maximizing the learning rate by responding to the quantitative measurement of learning, and (4) providing help for failure on the good answer rather than explaining bad answers (Anderson et al., 1995). When a tutor is in model tracing mode, there are three possible outcomes as the tutor tries to interpret each student action or sequence of actions:

1. The student's action matches one of the production rules from the domain model. (This will later result in an increment to strength in the quantitative model.)

2. Multiple sequences of production match, triggering a disambiguation question. (The disambiguation question would then control what productions are learned.)

3. There is no interpretation of the error. This may result in matching of a buggy production that leads to context specific messages for the student. If no buggy rule is matched, the student must continue to respond until a production is matched. Since this can cause an impasse, the system typically involves backup hints of greater and greater complexity to scaffold the student who is producing unmatchable responses.

There is a distinction between this model tracing process of interpreting behavior and the knowledge tracing process of inferring the growth of knowledge across the sequence of practice. Corbett later presented a canonical paper on what has become known as BKT, which included detailed modeling of individual student differences as well as the core Markov model knowledge tracing formalism (Corbett & Anderson, 1995). This augmented overlay model was used to track the student's progress from problem to problem by building a profile of strengths and weaknesses relative to the production rules (Anderson et al., 1995), rather than in relation to concepts or facts which had been tracked in previous domain overlay student tracing models. Step-by-step BKT is incorporated in a number of tutors in the PSLC (Aleven, McLaren, Roll, et al., 2006; Anderson et al., 1995; Heffernan, Koedinger & Razzaq, 2008; Ritter, Anderson, Koedinger & Corbett, 2007; VanLehn, 2006) and is a method used heavily by Carnegie Learning, Inc., to track skill progression in their various systems. Interestingly, however, many of the running systems do not realize the full complexity of Corbett's original models of individual differences (Corbett & Anderson, 1995).

A primary weakness of this type of approach for domains with many skills or misconceptions is that the skills and misconceptions must be reasonably enumerated in order to provide feedback (Brown & VanLehn, 1980). The creation of this domain model can be time consuming. Although development time is not widely reported, general ITS system design is estimated at 200–300 hours of development time per 1 hour of instructional content (Aleven, McLaren, Sewall, et al., 2006). Additionally, there is evidence that the remediation of specific bugs may not have added instructional benefit over the alternative of simply re-teaching the material that contained misconceptions (Sleeman, Ward, Kelly, Martinak & Moore, 1991). There is potential to mitigate this weakness with recent efforts to focus misconception database minimization, keeping the conceptual misconceptions to fewer than five in order to save development time (VanLehn et al., 2007). Sometimes, for well-defined domains with substantial prior data, the misconception or skill database may be automatically generated, saving a significant amount of development time (Burton, 1982; Cen, Koedinger & Junker, 2006). The GIFT domain module currently supports the communication of this information, but not the construction. Possibly, this type of knowledge component overlay could be accomplished automatically from the relationship and interactions inherent in textual information (Ahmed, Toumouh & Malki, 2012; Lahti, 2010). Practically, this type of knowledge overlay should either be supported via automatic generation, the merging of existing model structure, or a tool for authoring.

Perhaps more technically problematic than creating some list of skills is to describe in a meaningful way how the skills interact in a way that is computationally tractable and not overly simplistic. In a highly interactive system, the pedagogical model would be able to select items for practice that accounted for prerequisite relations automatically. We might want a model where we can make inferences about the item that is not yet mastered, but yet still the easiest item-type that the student might practice. For example, in a unit on fraction skills, we might want the system to detect (based on a history of success and failure) what basic skills a student has mastered (e.g., least common multiples) and avoid these, identify the skills students are ready for (e.g., equivalent fractions), and similarly hold in reserve those items that still have unmastered prerequisites. A related issue is when there are multiple skills needed to produce a single response. In such cases, whether we model the relationship of the skills as additive (one skill can compensate for others) or conjunctive (all skills must succeed to succeed in the response) has implications for parameter estimation and what skills to credit in the case of success or failure. However, in practice, it has been complicated to work out how skills should be combined (Ayers & Junker, 2006; Cen, Koedinger & Junker, 2008; Koedinger, Pavlik Jr., Stamper, Nixon & Ritter, 2011).

Constraint-Based Student Models

In constraint-based tutors, the domain model consists of a set of constraints representing the basic domain principles (Ohlsson, 1992). Each constraint is a declarative statement composed of a relevance condition (R) and a satisfaction condition (S). The relevance condition specifies when the constraint is relevant and only in these conditions is the constraint meaningful. The satisfaction condition specifies additional conditions that must be satisfied by the student's solution for which the constraint is relevant in order for the solution to be correct. A satisfied constraint corresponds to an aspect of the solution that is correct. A violated constraint indicates a mistake in the solution without explicitly representing the actual misconception the student might have; it simply means that the student's solution violates a domain principle. Therefore a constraint set explicitly defines the space of correct knowledge, without requiring misconceptions to be identified and collected; the space of incorrect knowledge is represented implicitly by violated constraints (Ohlsson & Mitrovic, 2007).

An example constraint from the cooking domain might be “when making French fries, the oil must be hot before potatoes are added.” The relevance condition of this constraint specifies the task (making French fries) and the current state of the solution (the student has put potatoes into the pan). The satisfaction condition then specifies that the temperature of the oil must be within the appropriate range. This constraint will be violated by various incorrect actions (like there is no oil in the pan, the oil is too hot or too cold, etc). However, the tutor only needs to teach the domain principle (the required temperature of the oil) to the student, rather than having to identify the exact misconception that caused the mistake.

One example constraint for the task of fraction addition might be relevant in situations when the student has added two fractions, a/b + c/d, and the student’s solution is of the form (a+c)/n (Ohlsson, 1992). The satisfaction condition of the constraint states that b=d=n; only in that case is the student’s solution correct. This constraint can be violated because of many misconceptions; however, the tutor can simply reinstate the violated domain principle (the numerators of the two given fractions can be added if their denominators are equal).

Constraints can be syntactic or semantic in nature (Mitrovic & Ohlsson, 1999). Syntax constraints are simpler than semantic ones, which need to ensure that the student solution is an appropriate solution for the given problem. In domains where there are multiple correct solutions, the required properties of the solution are captured in terms of a pre-specified (ideal) solution; the semantic constraints compare the student’s solution to the ideal solution taking into account alternative ways of solving the same problem, in terms of equivalent domain operators (Mitrovic, 2012).

Constraints specify what ought to be so, so they are evaluative in nature. They are not prescriptive in the sense of generating behavior like production rules do. Therefore, a student’s solution is diagnosed by matching it to the constraint set, identifying the relevant constraints, followed by checking the satisfaction conditions of the relevant constraints. This process corresponds to what VanLehn (2006) calls the inner loop. Information about satisfied and violated constraints is then used by the tutor to generate positive and negative feedback for the student (Mitrovic, Ohlsson & Barrow, 2013; Zakharov, Mitrovic & Ohlsson, 2005).

The previous two example constraints demonstrate that constraint-based modeling (CBM) does not presume any particular stepwise pedagogical approach. Instead, it is used to react to every action the student makes (to provide immediate feedback) or wait until the solution is complete and thereby provide delayed feedback. Usually feedback is provided on demand, when the student requires it, but can also be provided at times when the pedagogical model identifies that the student would benefit from the feedback. The granularity of constraints can also vary, and should be determined from the pedagogical point of view, by designing feedback that is effective for learning a particular task.

Student models in constraint-based tutors are also overlays. That is, the student’s knowledge is represented in terms of constraints that the student does or does not know, as demonstrated by the submitted solutions. Each time the student’s solution is matched to the constraints, information about violated and satisfied constraints is used to update the student model. The simplest way to represent the student’s knowledge of a particular constraint is a simple frequency of correct use within a specified window of opportunities to use the constraint. Some constraint-based tutors have probabilistic student models, where information about the violated and satisfied constraints is used to update the probability of the student knowing each constraint (Mayo, Mitrovic & McKenzie, 2000).

This overlay student model is used by the pedagogical module to select problems of appropriate complexity for the student. Problem selection could be based on simple heuristics, such as the one which identifies the constraint(s) with which the student had the most difficulty (Mitrovic, 2003). Alternatively, in one version of SQL-Tutor, there is a simple Bayesian network for each problem, which makes predictions about the performance of a particular student on the problem. These multiple predictions are then combined to give an overall measure of the value of the problem for a particular student (Mayo et al., 2000). CAPIT, an ITS that teaches children about punctuation and capitalization rules in English, uses a data-centric methodology, in which the structure of the Bayesian network is induced from the student data (Mayo & Mitrovic, 2001). The Bayesian model predicts the student's performance on each problem, and utility functions are used to select the best problem for the student and also the best feedback to be given to the student if there are errors. The classroom evaluation shows that such a decision-theoretic pedagogical strategy results in reduced learning times and improved performance.

Early research on constraint-based tutors focused on a range of domains to which CBM is applicable. Successful constraint-based tutors cover a wide variety of instructional domains, ranging from well-defined to ill-defined design tasks (Mitrovic & Weerasinghe, 2009). Well-defined tasks have crisp rules that govern them and often have incremental performance steps, while ill-defined tasks frequently have performance described in terms of outcomes. Some examples from the latter category are the SQL-Tutor (Mitrovic, 1998) that teaches database querying using Structured Query Language (SQL) and EER-Tutor that teaches database design using the enhanced entity-relationship (EER) model (Mitrovic, Martin & Suraweera, 2007). Learning gains have been between 0.6 and 1.6 sigma (Amalathas, Mitrovic & Ravan, 2012; Mitrovic, 2012; Suraweera & Mitrovic, 2004) after short learning sessions (less than 2 hours). Later work includes using constraints to represent not only domain knowledge, but also collaborative skills (Baghaei, Mitrovic & Irwin, 2007), to support tutorial dialogues (Weerasinghe, Mitrovic & Martin, 2009), self-assessment and open student models (Mitrovic & Martin, 2007).

The CBM paradigm, like the other types of student models in this work, can be used to represent domain knowledge. It has shown strength in its ability to instruct, but the largest limitation remains that that the models must still be authored. However, there is evidence that the authoring of constraint-based tutors is more rapid than their traditional counterparts (Mitrovic, Koedinger & Martin, 2003). These development times have come down rapidly through the introduction of authoring tools: from 220:1 (Mitrovic & Ohlsson, 1999) down to 30:1 (Heffernan, Turner, Lourenco, Macasek & Nuzzo-Jones). In addition, ASPIRE is an authoring system for constraint-based tutors that is freely available (Mitrovic et al., 2009).

CBM is an attractive solution to the student knowledge modeling problem for several reasons. Firstly, CBM does not require the large amounts of development time traditionally associated with production rules. It does not require probabilistic estimates of student knowledge built upon a large database of previous interactions, so it is a practical approach for instruction. Secondly, CBM supports both procedural and open-ended task modeling because it does not have to exhaustively model task dynamics (Mitrovic et al., 2003). Finally, CBM systems have been shown to be effective on a wide variety of instructional tasks (Mitrovic, 2012).

Yet another advantage of CBM is that the created models of domain knowledge have the potential to be transitioned to another system that embodies the same concepts. This is neatly aligned with the principles of architecture creation. The method of model creation thereby has the capability to transfer. The lifespan of the model is tied to the lifespan of the ITS, although fractional components of the model may be used in the construction of a model for another system. The ability to cannibalize model components and model construction components into a new ITS is a highly desirable architectural enhancement.

At the time of writing, GIFT supports the communication of student state information through a hierarchy of "concepts" and "sub-concepts," which have varying levels of grading. This fits well with the student modeling nature of evaluating the student model through mastered versus violated constraints. GIFT currently makes an effort to describe a student model using a concept breakdown, where each concept is not tied to a given domain of instruction, and is assessed simply (e.g., above standard, at standard, below standard, or unknown); however, it does not currently support the seamless authoring of module components which send these messages. The transfer of these tools and knowledge is the prime advantage of the CBM approach, but future development should support the plugin of the existing authoring tools to create CBMs for tracking student performance models.

Overall Considerations Underlying Overlay Models

Overlay models have two architectural components: the overlay objects and the mathematical tracking formalism. The overlay objects are typically linked to inner loop immediate pedagogy provided through the tutor's interface, e.g., immediate feedback when a constraint is violated, while mathematical formalisms are used by the outer loop to select which problem items are presented to students. Therefore, general systems like GIFT will benefit from providing explicit support for defining overlay objects that are linked both to tutor interface pedagogy and data recording that will enable the outer loop mathematical model to compute the next best problem to present at any time. While overlay models can function at one level or the other (e.g., with no immediate pedagogy or with no mathematical tracking), most modern overlay systems include both immediate pedagogy in response to violations of the assumption of the overlay objects and mathematical tracking of the overlay objects.

The strengths of the quantitative models underlying the overlays are not frequently realized in running systems because they typically do not live up to the goals of economic optimization and student-level individual difference tracking inherent in their configuration. Economic optimization takes account of the costs of each possible pedagogical action in addition to the gains (Atkinson, 1972a; Smallwood, 1962, 1971). While efficiency tends to be ignored, there has been renewed attention recently, with some researchers exploring how knowledge tracing models can be used to detect when skills are being over practiced, and instead spend that practice time on items that need practice. Reducing such over practice in existing systems has led to reduction in learning time, with no difference in gains (Cen, Koedinger & Junker, 2007). This failure to address costs in learner models is similar to the general failure to address student-level parameters in student modeling. Student-level parameters can be used for modeling individual differences, rather than making the assumption that there is a global, generalized model that fits all students. For example, despite work by Corbett showing that student-level parameters are important to produce tight fits to students' data (Corbett & Anderson, 1995), systems using BKT tend not to use subject level parameters, e.g., Carnegie Learning, Inc., tutors do not use general parameters for each student (Ritter & Anderson, 2006).

Because of the wide usage of component overlay models combined with mathematical tracing of components, GIFT will likely need to provide some support of this sort of student model. To start with, this implies support for a domain representation such as a Q-matrix, which assigns skills to each exercise in the domain. These Q-matrices may be universally needed for almost all overlay model variants. Secondly, GIFT needs to provide data structures to fully reload the students' prior history within a system to enable reconstruction of the user model. Third, GIFT should provide an API for mathematical functions that compute pedagogically relevant quantities using the history for the student. For example, in the Fact and Concept Training system, the mathematical model includes a function "choose-trial," which analyses the full history for all the items for the student to determine the item that is closest to being at the optimal point for rehearsal. In turn, this choose-trial function (called from the interface) depends on lower level model computations of the probability of successful performance and time costs for each item. Similarly, after practice, a model API function saves the result of prior practice as the history accumulates (Pavlik Jr. et al., 2007).

We advocate a three-pronged recommendation that the domain structure should be compatible with a Q-matrix structure (Domain Module), able to reload the history of practice so as to compute the current student model state (Student Module), and an interface for interpreting mathematical states of the student model to make pedagogical inference in a tractable way (Pedagogical Module). If a basic model API that includes some default models, GIFT users will be able to test and modify the mathematical component of the overlay models quickly and easily. Additionally, by providing a standard Q-matrix support in GIFT for the overlay of skills as a separate component of the architecture (a different API), it should be possible for users to test different mathematical formalisms to capture in the context of the same Q-matrix. This flexibility should speed development by modularizing the Q-matrix and mathematical tracing individually, thus allowing easier reuse/generalization of Q-matrices or mathematical models. By association, adherence to this recommendation will yield increased learning gains with decreased development times.

Knowledge Space Models

The domain model of knowledge space theory is a large number of possible knowledge states in a domain (or knowledge structure), whereas the student model is a record of which of the knowledge states are mastered. It is essentially a fine-grained overlay model that is not based on cognitive structures (e.g., skills), but is rather based on problem types that are or are not mastered. A student's competence is reflected in the student model as a probabilistic estimate of the types of problems that the student is capable of solving in the domain (Falmagne, Doignon, Cosyn & Thiery, 2003).

The knowledge structure in knowledge space theory is based on the precedence relation, in which ability to solve one type of problem tends to precede solution of another type. Such precedence relations may be due to the prerequisite structure of the domain, but also may be due to the order in which problem types are currently taught (Falmagne et al., 2003). These precedence domain models are then used in a tutoring context to select the next problem to work on that is sensitive to the student's competence as tracked by the overlay student model of the preceding problems currently known. Thus, depending on all the problem types previously mastered in the student model, there is "just prior" (inner fringe) and a "next subsequent" (outer fringe) of items in the problem. The outer fringe constitutes the pedagogical decision of the knowledge space student model given current the probabilities of being in the various knowledge states as inferred by the prior success on individual problem types (Falmagne et al., 2003).

A knowledge structure in knowledge space theory can be described as a Bayesian network without hidden nodes, since each of the nodes maps to a concrete class of problems (Desmarais & Pu, 2005). Knowledge spaces are often hand engineered because of the massive amounts of data needed to infer the complete graph of the domain knowledge precedence relations but such structures also need to be refined with data to enhance accuracy (Falmagne et al., 2003). Part of the reason for the large amounts of data that are needed is because of the extraordinary flexibility of the knowledge space formalism to represent AND/OR precedence relations in which the outer fringe can be arrived at through one or more pathways (Desmarais, Maluf & Liu, 1996). In other words, C can be known if either A and/or B are known.

Partial Order Knowledge Structures (POKS) have been developed as an alternative to account for the difficulties of inferring the AND/OR structure from data alone, as reviewed elsewhere (Desmarais et al., 1996). As an alternative to the AND/OR graphs of the knowledge structure, POKS use a simpler precedence relation that is less powerful but easier to estimate since it requires that all of the prior states be present. Therefore, it captures an AND graph that can be laid out as a directed acyclic graph (DAG) in which each problem type requires all prior problem types to be mastered with some threshold of certainty. Arguments have been made that such a formalism is adequate in part because situations in which there are alternative prerequisites can be quite rare (Desmarais et al., 1996). Conveniently, explicit mathematical examples and explanation on how to construct knowledge spaces using POKS are detailed in the literature (Desmarais et al., 1996; Desmarais & Pu, 2005).

The best example of a fully deployed knowledge space based system is the ALEKS mathematics tutor (Doignon & Falmagne, 1999; Hu et al., 2012). In this system, there is a very large knowledge space model for mathematics. The system begins with a student model building phase in which it asks questions without pedagogical intent, but rather to determine the student state. This assessment process is also typically set to run after 5 hours or 20 item types are passed so as to recalibrate the state of the student model. After the student model is determined (the state of the student relative to the domain representation, i.e., the inner fringe), the outer fringe can be inferred from the domain structure. This inference of the outer fringe appears to be the primary pedagogical move in the ALEKS system. The pedagogical model is simply to give the student the choice of any problem type in their outer fringe. Once working on a problem type the student model is straightforward, because it simply involves allowing the student to branch to a new outer fringe problem type if they get the current problem type correct four times in a row, though they have the option of more practice if they wish (personal communication, ALEKS Inc.; http://www.aleks.com/about_aleks/research_behind).

While some ITSs focus on context sensitive adaption either through constraints or skill models, knowledge space theory systems (as exemplified by ALEKS) rely on a far more complex domain model. This domain model essentially presolves the appropriate possible branches from one state to another. This presolved domain-space solution maps to pedagogy through the detailed precedence relationships, leaving the system very little "intelligence" in deciding content order. This methodology harkens back to the branching systems discussed earlier. Essentially, ALEKS is a controlled branching network system with limited student choice of options from the outer fringe of the network. Importantly, knowledge space systems have not made a great effort to consider the different possible branching rules that might determine which outer fringe item is given to a student (this is left in the student's hands) or how many repetitions may actually be needed to transition from one type to another. So, while knowledge space student models provide a nice example of branching, this student modeling method, while parsimonious, could be made more flexible if the permitted pedagogical inferences were allowed to be more complex as a function of the student's history of performance.

Systems other than ALEKS have also been developed, most notably the work on the Catalyst system for chemistry (Arasasingham, Martorell & McIntire, 2011; Arasasingham, Taagepera, Potter, Martorell & Lonjers, 2005). This system seems to be quite effective, but relies on a strategy of breaking the knowledge space up into different representational foci, including numeric, symbolic, and visual problem types so as to be able to plan learning precedence relations that include attention to integrated representations of the content for the students. We might speculate that this improves effectiveness based on research on multiple representations, making it difficult to assess the pure contribution of the model. Student modeling with knowledge spaces appears to be a fertile area to the point where other independent groups are contributing significantly to development of the techniques (Albert & Lukas, 1999).

Intelligent tutoring frameworks like GIFT should certainly include support for knowledge space student modeling. Knowledge space student modeling uses the structure of the domain (perhaps represented by a matrix of problem to problem relationships) to assess where the students are in that domain (the student model). In GIFT the Q-matrix formalism might be expanded to include matrices that encode precedence relations, perhaps by referring to the matrix as the "domain map." The student model using knowledge space computes a likelihood across the possible states of the domain map based on some prior assessment. While the procedure of updating the likelihood seems like it will be conceptually complex, it also appears analogous to the process of inference in the overlay models that results in an update to their long-term model quantities. In both cases (knowledge spaces and overlay models), the system makes pedagogical inferences based on some numeric representation of the long-term student model state.

Dialogue Student Models

This type of student modeling is typical for ITSs that help students learn by holding a conversation in natural language, such as AutoTutor or Why-Atlas (Graesser, D'Mello, et al., 2012; VanLehn et al., 2007). Although these conversational ITSs based in natural language vary in terms of the pedagogical activities they support, they all share two defining attributes. First, these conversational ITSs are based on naturalistic observations and computational modeling of human tutoring strategies embedded in tutorial dialogue (D'Mello, Olney & Person, 2010; Graesser & Person, 1994; Graesser, Person & Magliano, 1995; Person, Graesser, Magliano & Kreuz, 1994). A common strategy is the so-called five-step dialogue frame (Graesser & Person, 1994; Graesser et al., 1995; Person et al., 1994): (1) Tutor asks a deep reasoning question, (2) Student gives an answer, (3) Tutor gives immediate feedback or pumps the student, (4) Tutor and student collaboratively elaborate an answer, and (5) Tutor assesses the student's understanding. This strategy reflects the other defining attribute of conversational ITSs, namely, the emphasis on collaborative, constructive activities based in theories of learning and tutoring (Aleven & Koedinger, 2002; M. T. H. Chi, 2009; M. T. H. Chi, Siler, Jeong, Yamauchi & Hausmann, 2001; Fox, 1993; Graesser et al., 1995; Moore, 1994; Shah, Evens, Michael & Rovick, 2002; VanLehn, Jones & Chi, 1992).

Because conversational ITSs are fundamentally rooted in dialogue, their corresponding student models are usually structured in dialogue-centric terms. A useful (though distinctly non-ITS) oriented framework for modeling dialogues has previously be established by McTear (2002). This framework identifies three major kinds of dialogue models: graph-, frame-, and agent-based. In a graph-based system, at any given moment there are a fixed set of alternative user actions; analogously, each user possible input is associated with an arc from a given node/state of a graph to a new node/state. A common example is a phone-menu dialogue system, e.g., "Press 1 for directory, 2 for billing, or 0 to speak with an operator." Graph-based systems are relatively rigid because they restrict the user's input. Analogously, one would have to slowly progress through a phone-menu dialogue to get to the desired state, rather than being able to say, "I need to speak to billing" at the first opportunity. Graph-based systems are largely comparable to branching student models in terms of the tutor behavior they support. Frame-based systems are slightly more flexible than graph-based systems. In a frame-based system, there is a single overall goal represented by a frame, a fixed attribute-value data structure. The goal of the system is to fill each of the value slots in the frame, in any order. For example, an online reservation system might have the slots [*departure date, departure time, destination city, departure city, arrival time, arrival date*]. An initial dialogue move from a frame-based system might be, "What reservation would you like to make today?" If a user says, "I'd like to go to Vegas tomorrow," the system would recognize and fill two of the needed slots in the frame, and follow up with a question like, "What time would you like to leave tomorrow?" Thus, if the user gives a full frame's worth of information on the first turn, the system would not need to ask any more questions. If the user gives some but not all of the information, the system would flexibly request the missing information. Frame-based systems are loosely comparable to constraint-based models in terms of the tutor behavior they support. Each slot in a frame specifies a constraint that must be met in order for the frame to be complete. Agent-based systems are the most complex of the three types of dialogue system identified by McTear. The hallmark of agent-based systems is that they are fully mixed-initiative: the user can introduce new topics and goals for the system. This often (but not always) requires that the system be able to recognize the intentions of the user. For example, if the user is trying to find the area under a curve and says, "I don't know how to do integration," the system would (1) recognize that the current goal requires integration, (2) recognize that the user is unable to perform integration (a subgoal of the current goal), and (3) attempt to help by, for example, explaining integration to the user. This is a stereotypical example of an agent-based system; however, there is substantial variation in how much artificial intelligence (AI) and symbolic modeling is actually implemented in such a system.

While the three kinds of dialogue modeling defined by McTear could theoretically be applied in a pure fashion, in practice dialogue systems and ITSs blend elements of these three models, to varying degrees, depending on the functionality of the system. Each of the three provides different functional properties. Graph-based systems provide highly specific and state-dependent actions, are rigid in structure, and are robust computationally because user input is unambiguous. Frame-based systems provide less specific, context-free actions and are a flexible means of pursuing a single goal, though they are less computationally robust than graph-based systems because user input is less constrained. Agent-based systems are theoretically the most flexible because they allow unconstrained user input and interaction patterns, but in practice they are computationally fragile because the AI behind agent-based systems is at or beyond the current state of the art.

To better explain how these dialogue models interleave with student models in a conversational ITS, we give a detailed example based on AutoTutor. However, similar examples may be given for a variety of conversational ITSs including Guru, GnuTutor, and Operation ARIES (Millis et al., 2011; Olney, 2009; Olney, Person & Graesser, 2012). The overall structure of an AutoTutor session is as follows:

Problem statement→ Pump → **Expectation coverage*** →Summary

In other words, each session begins with an introduction ending with a problem statement. When the student responds, the tutor typically follows with a *pump*, e.g., “What else can you say?” Then the tutor launches zero or more cycles of expectation coverage, and follows these with a summary of the answer to the problem. From the above description, the overall structure of the dialogue is graph-based, with no alternatives. Correspondingly, at this level, there is no real need for student modeling. However, *expectation coverage* is not an automatic action, but rather a composite action or *mode* of the tutor; this is where nearly all of the tutor-student interaction takes place.

AutoTutor’s expectation coverage is modeled after *Expectation and Misconception Tailored* (EMT) dialogue found in authentic tutoring sessions (Graesser, D’Mello, et al., 2012). Human tutors typically have a list of expectations (anticipated good answers, steps in a procedure) and a list of anticipated *misconceptions* associated with each problem statement. For example, expectations E1 and E2 and misconceptions M1 and M2 are relevant to the following example physics problem:

> PHYSICS QUESTION: If a lightweight car and a massive truck have a head-on collision, upon which vehicle is the impact force greater? Which vehicle undergoes the greater change in its motion, and why?
>
> E1. The magnitudes of the forces exerted by A and B on each other are equal.
>
> E2. If A exerts a force on B, then B exerts a force on A in the opposite direction.
>
> M1: A lighter/smaller object exerts no force on a heavier/larger object.
>
> M2: Heavier objects accelerate faster for the same force than lighter objects

During expectation coverage, the goal of a conversational ITS is to elicit an explanation from the student with regard to the problem statement or seed question. Since student explanations often contain multiple propositions or sentences, this goal reduces to eliciting each sentence in the *expected* explanation. We label these subgoals expectations, because they are expected parts of the overall explanation. The similarity with frame-based systems should be evident: during expectation coverage, the goal of the ITS is to fill out a frame corresponding to the complete explanation to the initial problem statement. Each sentence of the explanation corresponds to a slot in the frame. The task for the ITS is to elicit each expectation in the frame, which involves two subtasks. First, the ITS must choose a strategy that results in a student response matching an unsatisfied slot/expectation. Second, the ITS must evaluate the student response to determine which (if any) of the unsatisfied slots/expectations it satisfies. Thus, the student model during expectation coverage is largely a frame type structure representing what (if any) of the expectations the student “knows.” The ITS uses this information to focus on slots the student does not know, creating a more adaptive and efficient learning experience. Slots may be selected using criteria such as semantic proximity to the current slot (i.e., zone of proximal development).

However, for each slot/expectation, the student model may be augmented with a measure of the student’s ability, based on the strategy the tutor used that resulted in a satisfied slot/expectation. The connection between tutor strategy and student ability is based on the observation that some strategies require more *effort* from the student in terms of recall and cognitive processing (e.g., prediction, inference, causal reasoning). A major strategy is asking questions ranging from vague to highly specific (Graesser et al., 1995; Person & Graesser, 2003). For example, a pump is a vague question consisting of neutral back channel feedback (uh-huh, okay, subtle head nod) or explicit requests for more information (what else, tell me more). Pumping serves the functions of exposing student knowledge and encouraging students to construct content by themselves. Hints give context to the question, but do not lead the student to a specific answer, for example, “What about blood pressure in this situation?” In contrast, prompts are highly specific questions in which tutors supply a discourse context and prompt the student to fill in a missing word or phrase, for example, “As the heart beats faster, the blood pressure does what?” In addition to pumps and prompts, many other question types exist that vary in specificity and depth of processing required by the student, such as disjunctive questions or causal questions (Graesser & Person, 1994). For a given expectation, if a student generated an answer satisfying it when the tutor strategy was more difficult, then the student may have greater ability/mastery of that expectation than if the strategy was easy, for example a prompt. Thus, within an expectation coverage cycle, the point at which a student satisfies the expectation is another aspect of the student model. In AutoTutor, the expectation coverage cycle is as follows:

$$\text{Hint} \rightarrow \text{Prompt} \rightarrow \text{Assertion}$$

This provides two levels of mastery assessment, since a tutor *assertion* is a paraphrase of the expectation given to the student. This cycle is graph-based, but the cycle terminates when the expectation is satisfied.

As mentioned previously, a conversational ITS not only uses strategies to elicit a student response but also evaluates the student response to determine if it satisfies any expectation. In addition, a conversational ITS typically anticipates student misconceptions. An expectation or misconception is scored as being expressed by a student if the student articulates it in natural language with a high enough semantic match. Semantic matches can be assessed by a number of methods in computational linguistics, such as content word overlap, latent semantic analysis, regular expressions, semantic entailment, or Bayesian statistics (Cai et al., 2011; Graesser, Penumatsa, Ventura, Cai & Hu, 2007; Rus, McCarthy, Graesser & McNamara, 2009; VanLehn et al., 2007). Depending on the method used, the match between an expectation/misconception and the student response can range from a single number between 0 (no match) and 1 (identical), or can consider the response compositionally in order to debug the answer. In either case, the evaluation becomes part of the student model.

If the evaluation is a single number, then the frame becomes a “soft frame,” where expectations are satisfied if the evaluation is above a numeric threshold. Even when the expectation is satisfied, the evaluation score can be used to indicate relative mastery. For example, if the evaluation score is above the threshold of 0.5, then the ITS may decide that the corresponding expectation is sufficiently “known” that the tutor can move on to other expectations. However, while evaluation scores of 0.6 and 0.9 would both satisfy the threshold and result in the same tutor behavior, the latter student may have mastered the expectation to a higher degree than the former. Thus, the student model in this case is a soft frame where each slot is weighted by the final evaluation score that led to the satisfaction of the expectation. Complementary extensions to the student model include the entire history of evaluation scores for each expectation. Even though these single number evaluations may appear one-dimensional, as part of the student model they may be used to generate complex adaptive behavior by the ITS. For example, a conversational ITS will usually select a specific question in order to maximize the chance that a student will satisfy an expectation (assuming the student generates the correct response). Likewise, a conversational ITS will usually select the next expectation to cover based on its similarity to the current expectation.

The advantage to a numeric semantic match that it is very robust to noise. The standard approach is to use latent semantic analysis (LSA), an unsupervised technique that creates numeric vector representations for words in a large unstructured collection of texts (Landauer, McNamara, Dennis & Kintsch, 2007). The so-called LSA vector space, together with the expectations and misconceptions described above, represents the domain model for an AutoTutor session. The corresponding semantic match in LSA is the vector cosine. This approach is very robust to ungrammatical and fragmentary user inputs and requires no knowledge engineering expertise. LSA type approaches are generally applicable to natural language inputs in domains where the order of words in the student's answer have little impact on the meaning of the answer. Counterintuitively, this order-free property applies in most cases (Landauer et al., 2007), though would not apply in cases where the answer contained mathematical formulae or similar representations.

As mentioned above, agent-based systems support the most sophisticated dialogue behavior but come with the cost of added complexity and loss of robustness. While generally true, it is possible to satisfice the problem of mixed-initiative in such a way that avoids the larger issues addressed by agent-based systems. In some versions of AutoTutor, mixed-initiative dialogues are created by recognizing and responding to student questions. Every student input is first analyzed to determine if it is an answer to a current tutor question or is a new student question, using a speech act classifier (Olney et al., 2003). When AutoTutor answers a question, it enters a sub-dialogue that is nested in the larger dialogue of the tutoring session. In essence, AutoTutor temporarily suspends its tutoring agenda in favor of answering the question posed by the student. Different student question types indicate shallow (e.g., verification or definition) or deep level reasoning (e.g., causal consequence or inferential) and so can be used to inform the student model (Graesser & Person, 1994). However, while question asking is an important part of active learning, students tend to ask very few questions, even when specifically asked by AutoTutor, "Do you have any questions?" Thus, the utility of question classification and corresponding mixed-initiative dialogue may be low relative to the practical complexity of creating such a system. In addition, the potential of misclassifying a student input increases with the number of speech act categories, meaning that part of the cost of having a question-answering component is incorrectly classifying student answers as questions.

Dialogue-based student models have several advantages that make them attractive for a domain-independent, general purpose framework like GIFT. Numeric semantic matches (e.g., LSA) are highly robust when used to evaluate student input. Students can use fragmentary language or synonyms and still have their answers accepted by the tutor. However, numeric semantic matches are also prone to biases implicit in the corpus of texts used to create them. These biases can be invisible to authors who are creating dialogues for the tutor. For example in biology, the terms "prokaryotic" and "eukaryotic" are often mentioned in the same paragraph but compared and contrasted with each other. Because of this, a biology LSA space will give a high numeric semantic match between the two, even though they are quite different (eukaryotes have a nucleus and membrane-bound organelles, e.g., animals and plants, and prokaryotes do not, e.g., bacteria).

Because of this problem (and biases in semantic spaces more generally), we propose the following for GIFT as well as future work that uses numeric semantic matches in dialogue student models. First, the semantic space should be developed before the dialogues are created and incorporated into the dialogue authoring tool. Second, when the expectation and misconception dialogues are created, the authoring tool should use the semantic space to evaluate the dialogue as it is authored and offer real-time feedback. For example, suppose that the author has just created a prompt, "What do we call cells that have a nucleus?" with the correct answer "eukaryotic cells." If the authoring tool lists the near neighbors of "eukaryotic cells" in the semantic space, then the author will become aware that an incorrect answer, "prokaryotic cells," would be accepted as a correct answer for this prompt. This gives the author the opportunity to apply additional constraints, like regular expressions, to ensure that only correct answers will be accepted (Graesser, D'Mello, et al., 2012). This kind of feedback will help keep the student model from being too forgiving of student input.

A second kind of real-time feedback that we propose involves checking the correspondence between the expectation and the ideal student responses to questions involving that expectation. For example, if the expectation is, "Eukaryotic cells have a nucleus," with only the associated prompt "What do eukaryotic cells have that contains DNA?" then it's possible that the student could type "nucleus" but not actually cover the expectation. In other words, it's possible to write dialogue that would never guide the student to cover the expectation, leading to a malformed student model. A solution to this is to provide real-time feedback that computes the semantic match between all of the ideal student answers to the hints and prompts associated with an expectation and the expectation itself. If the match is very low, this could indicate a problem. Moreover, the threshold for the expectation itself is probably best determined by some percentage of this score rather than being held constant for all expectations in a problem. Following this recommendation should prevent the student model from being too harsh with student input.

State and Trait Identification Models

This category includes student models that capture affective/motivational constructs that are either transient or trait like. This category is distinct from overlay models that capture some sort of learnable skill (e.g., self-regulation). Since traits tend not to change (unlike skills) and states tend to fluctuate in reaction to stimuli (unlike skills), this category logically excludes any learned proficiency based models. Despite this, it seems likely that state and trait models would be used in conjunction with learned proficiency models as an additional input to infer student states and make pedagogical decisions.

Because of the role of affect and cognition in the learning process, a significant amount of ITS research is focused on measurement and detection (Cerri, Clancey, Papadourakis & Panourgia, 2012). This occurs through the incorporation of hardware sensors of software detectors into the standard learning system. Examples of these include an electroencephalography (EEG) head-cap (Goldberg et al., 2011), AI-based software models (Wixon, Baker, Gobert, Ocumpaugh & Bachmann, 2012), or a combination of both (Kapoor & Picard, 2005).

Human tutors are known to be as devoted to the motivation of the student as much as their cognitive and informational goals (Lepper & Hodell, 1989; Woolf, 2008) Affective characteristics have also been highlighted in the literature as an important area to learning (Woolf, 2010). While there has been significant research in their development, there has been limited transition to usable systems. One nice example of a strong success comes from feature detection in spelling learning (Baschera, Busetto, Klingler, Buhmann & Gross, 2011).

While this category has few example contexts where there are significant benefits to learning in a running ITS, the category has distinct potential, particularly as machined learning and sensor technologies develop to the point of practical application. The state and trait distinction has particular relevance to the architecture of general systems such as GIFT. In particular, GIFT should be able to handle stable student parameters that feed into the student model as individual differences. For instance, a prior assessment of each student's self-efficacy could be passed to the dialog based student model which would lead to the pedagogical inference to adjust the scripts to be more encouraging with low efficacy students. Similarly, sensor modules data would feed into student models in a continuous fashion so that it could be incorporated into any long-term student model (e.g., BKT) and thus affect pedagogical decision making (e.g., D'Mello & Graesser, in press).

Discussion

An ITS functions by having a domain model which defines the space of possible student states. The student model keeps tracks of each individual student within this possible state space. The pedagogical model is a set of rules that define when interventions are triggered as a function of the student state and the tutor interface. The student model represents the student state as a function of prior actions, either tracking these prior actions dynamically so that prior states are not necessarily stored or by keeping track the history of prior practice in order to compute student prior knowledge as the context demands. Pedagogical rules aim to maximize a measure of learning amount or efficiency.

We began our discussion of student models in the late 1950s with the invention of branched programming. Branched programming addresses student state differences by applying different pedagogical moves at fixed branch points afforded by the domain model. Branching models are also discussed because *fundamentally* all student models "branch" in the sense that any pedagogical move we might take is in essence a branch in response to the student state as encoded in the possibilities expressed by the domain model. These branches or adaptive choices must depend on something, and the student model state is the independent input (derived from student actions) that leads to the inferences about adaptive pedagogy. However, a universal branching system in the context of GIFT, or more generally, as how to use them as an effective generic design pattern for tracking student states to make rich and powerful pedagogical moves, is currently unclear.

VanLehn helps provides a design pattern for us to consider in his proposal about the behavior of tutoring systems where he proposed inner and outer loops to characterize common ways that ITSs have been configured (VanLehn, 2006). In this taxonomy, the inner loop was construed as the individual problem, which would be intelligently tutored in small cycles of interaction between the student and ITS as the student works on the problem. This is also known as microadaption. In contrast, the outer loop was described as a problem selection loop, from which a problem selection algorithm might choose the next item based on higher criteria than behavior at the single problem level. This is known as macroadaption. Essentially, this is a hierarchy of kinds of branching, with the inner loop often representing stereotyped branching schema within problems for individual steps, and the outer loop specifying sequencing/ branching between exercises. Of course, within this hierarchy we might describe more levels. For example, Pavlik Jr. & Toth (2010) proposed an additional level of branching complexity, the curriculum loop, which assumes that a student model would track states for broad categories of problems (units of content) and would allow pedagogical models to execute actions at an additional level, as in the case of sending a student into a basic skills review unit if repeated arithmetic errors are made in an algebra unit.

In mature ITS, there is a blend of student models. For example, in cognitive tutors, problem selection occurs with some version of BKT (Corbett & Anderson, 1995) used as a student model for the outer loop, but within problems the student model tends to be more like a constraint-based model (even though often represented as production rules) with inner loop pedagogical responses to student actions disconnected from the outer loop BKT student model. For instance, error flagging or on-demand hints to the student represent the within problem student model as a collection of response constraints. These “should” statements define what features of the interface need to be modified in what way to solve the problem, while the pedagogical model specifies what inference to make based on the current state of the interface.

The cognitive tutor example illustrates how a learning system could potentially have a student model that allows state branches at multiple levels of the domain (problem step, overall problem, curriculum unit), where the means of modeling the branching at each level of the student model may be different due to the pedagogical needs of that level of the domain. Similarly, we could imagine a ITS with a constraint-based student model for tutoring algebra problem solving as the inner loop, a knowledge space domain model used for the outer loop problem selection, and a curriculum branching loop that that branches students into a basic skills unit if arithmetic constraints are not met. The second arithmetic outer loop might use an overlay model for all the facts in the times tables, with no inner loop for these simple facts. This discussion places further constraints on GIFT by illustrating the advantage of having a hierarchical organization of student models for the multiple levels of domain structure so as to facilitate maximum flexibility for pedagogical strategies at these multiple levels.

Figure 5-1 shows the three levels at which pedagogy occurs that seem important to allow in GIFT. The figure attempts to convey the relationship between the three levels, in that the inner loop, which itself represents a collection of states and consequent pedagogical moves) is entered from the outer loop problem. Similarly, an outer loop sequence of problems is selected as the pedagogical decision from the curriculum student model. A simulation scenario example should help clarify. If the curriculum involves practicing team search and reconnaissance missions, we might suppose the curriculum loop would be a number of different contexts for reconnaissance, such as night missions, search and recovery missions, rough terrain missions, etc., each of which of needs to be completed with a certain criterion (according to the student model) before the next context is selected (according to the pedagogical model). Each outer loop problem might be viewed as a specific scenario within a context (e.g., rescue mission in Mogadishu) that itself has to be mastered or repeated, according to the decisions of the outer loop. Finally, within each scenario (the inner loop in this context) we might expect that there would be specific tasks, each with their own states for the student that warrant specific pedagogies.

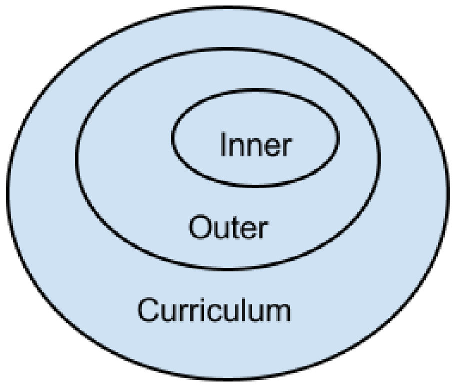


Figure 5-1. Pedagogical levels at which a student model may be used to make decisions

This discussion is useful because it helps set the stage for considering the specific usefulness of particular student models. Student models may generalize to content, but may not as easily generalize to different pedagogical levels. So, for instance, dialogue-based tutoring seems ill suited to handling the problems of curriculum sequencing or outer loop management (e.g., problem selection). Similarly, selection of steps based on some mathematical model makes little sense since the steps of a problem are more or less fixed once that problem is selected. Table 5-1 attempts to summarize our assessment of the student models we have reviewed.

Table 5-1. Summary evaluation comments

	Quantitative Fit	Ease of Understanding	Generality and Flexibility	Cost of Creation	Granularity	Time Scale	Learning gains in practice
Programmed Branching	N/A	High	Low	Low	Any	Short-term	Depends on adequacy of specific domain model
Rule Space	High	High	High	Low	Any	Short or Long-term	Little direct evidence
Model Tracing	High	Low	Moderate	High	Production rules	Potential long-term but not always realized	Yes, strong gains for particular domains
Constraint-Based Tutors	High	Moderate	High	Moderate	Any	Potential long-term but not always realized	Yes, strong gains for particular domains
Knowledge Spaces	Good at item-category level	Low	Low	High	Item-categories	Long-term	Yes, strong gains for particular domains
Dialogue	N/A	Moderate	Moderate	Moderate	Inner loop	Short-term	Yes, strong gains for particular domains
State and Trait	Generally low	Moderate	Unclear	Moderate	Any	Short- and/or long-term	Unclear

Conclusions

There is significant challenge in developing a standard, or framework, to support an existing field. The goal behind developing standards is that they will be adopted, which can only happen if the individual adopters find the new standard useful to their practice. As such, it is important to discuss the benefits of adopting such standards.

GIFT will enable new ITS creators to experimentally test their methods of student model creation with differing domains of instruction, pedagogical strategies, and ways of assessing user states through sensors. GIFT will enable the experimental comparison of one type of model against another, while keeping the other portions of the ITS exactly the same, thus enabling good unconfounded comparisons in the ITS. GIFT will enable quicker construction of ITS application through the ability to standardize user tools for student model construction.

However, these advantages come at a price. This is the price that all standards impose on the community of adoption. At the simplest level, conforming to a “Phillips” screw-head standard destroys the potential for the creation of alternate screw-head designs. No one framework, standard, architecture, or toolset can accommodate for all possible solutions. Despite this, to promote adoption, the created solution should support as many of the current practices as is possible. With these tradeoffs in mind, we conclude with a few points about the process of developing GIFT.

The first point is that the standard approach to instruction remains simple branching programs with multimedia content. Therefore, most student models are merely a record of the place in such a branching program structure. Such systems are still important because intelligent tutoring is not required for many kinds of instruction, such as first-time information presentation. This first assumption leads to the first recommendation: GIFT should support authoring of simple branching programs.

The second point is that the most mature and applicable forms of student modeling are those which have had the most research and software development effort. These are the methods for adaptive instruction that have been empirically proven to work, have multiple research teams working on them, have deployed ITSs to classrooms, and have developed authoring tools to speed developments and research. GIFT, in order to support successful adoption, needs to be inclusive of these models. Becoming inclusive of these models implies that GIFT becomes inclusive of the tools that are used to create them. This second implication is that GIFT should support the importation, with very minimal change, of student models and authoring tools created within CBM, model tracing, and dialog-based paradigms for inner loop mediation. They have proven to be successful and widely adopted.

A third point is that several of the methods, most notably overlay and knowledge spaces, employ “domain maps” that characterize the tasks and the relationships between these tasks. Because of this commonality, it seems likely that GIFT would benefit from providing such a domain map data structure that could be referenced when making pedagogical decisions in reference to the student model. Further, such domain maps are typically connected to a long-term computational student model that mathematically traces the probability of the tasks relative to the prior progress of the student. This implies GIFT will be well served by providing the capacity to mathematically compute pedagogically relevant student model quantities at any time by processing the student’s prior data (which is encoded relative to the domain map).

References

Ahmed, K. B. S., Toumouh, A. & Malki, M. (2012). Effective Ontology Learning: Concepts’ Hierarchy Building using Plain Text Wikipedia. In *Proceedings ICWIT* (pp. 171).

Albert, D. & Lukas, J. (1999). *Knowledge Spaces: Theories, Empirical Research, and Applications*: L. Erlbaum.

Aleven, V. & Koedinger, K. R. (2002). An effective metacognitive strategy: Learning by doing and explaining with a computer-based Cognitive Tutor. *Cognitive Science, 26*, 147–179.

Aleven, V., McLaren, B., Roll, I. & Koedinger, K. R. (2006). Toward meta-cognitive tutoring: A model of help seeking with a cognitive tutor. *International Journal of Artificial Intelligence in Education, 16*, 101-128.

Aleven, V., McLaren, B., Sewall, J. & Koedinger, K. (2006). The Cognitive Tutor Authoring Tools (CTAT): Preliminary Evaluation of Efficiency Gains. In M. Ikeda, K. Ashley & T.-W. Chan (Eds.), *Intelligent Tutoring Systems* (Vol. 4053, pp. 61-70): Springer Berlin / Heidelberg.

Aleven, V., McLaren, B. M., Sewall, J. & Koedinger, K. R. (2009). A New Paradigm for Intelligent Tutoring Systems: Example-Tracing Tutors. *International Journal of Artificial Intelligence in Education, 19*, 105-154.

Amalathas, S., Mitrovic, A. & Ravan, S. (2012). Decision-Making Tutor: Providing on-the-job training for oil palm plantation managers. *Research and Practice in Technology-Enhanced Learning, 7*, 131-152.

Anderson, J. R., Conrad, F. G. & Corbett, A. T. (1989). Skill acquisition and the LISP tutor. *Cognitive Science, 13*, 467-505.

Anderson, J. R., Corbett, A. T., Koedinger, K. R. & Pelletier, R. (1995). Cognitive tutors: Lessons learned. *Journal of the Learning Sciences, 4*, 167–207.

Arasasingham, R. D., Martorell, I. & McIntire, T. M. (2011). Online Homework and Student Achievement in a Large Enrollment Introductory Science Course. *Journal of College Science Teaching, 40*, 70-79.

Arasasingham, R. D., Taagepera, M., Potter, F., Martorell, I. & Lonjers, S. (2005). Assessing the Effect of Web-Based Learning Tools on Student Understanding of Stoichiometry Using Knowledge Space Theory. *Journal of Chemical Education, 82*, 1251.

Atkinson, R. C. (1972a). Ingredients for a theory of instruction. *American Psychologist, 27*, 921-931.

Atkinson, R. C. (1972b). Optimizing the learning of a second-language vocabulary. *Journal of Experimental Psychology, 96*, 124-129.

Atkinson, R. C. & Crothers, E. J. (1964). A comparison of paired-associate learning models having different acquisition and retention axioms. *Journal of Mathematical Psychology, 1*, 285-315.

Ayers, E. & Junker, B. (2006). Do skills combine additively to predict task difficulty in eighth grade mathematics? In J. Beck, E. Aimeur & T. Barnes (Eds.), *Educational Data Mining: Papers from the AAAI Workshop* (pp. 14-20). Menlo Park, CA: AAAI Press.

Baghaei, N., Mitrovic, A. & Irwin, W. (2007). Supporting collaborative learning and problem-solving in a constraint-based CSCL environment for UML class diagrams. *International Journal of Computer-Supported Collaborative Learning, 2*, 159-190.

Baker, R. S. J. d., D'Mello, S. K., Rodrigo, M. M. T. & Graesser, A. C. (2010). Better to be frustrated than bored: The incidence, persistence, and impact of learners' cognitive–affective states during interactions with three different computer-based learning environments. *International Journal of Human-Computer Studies, 68*, 223–241.

Barnes, T. (2005). *The Q-matrix Method: Mining Student Response Data for Knowledge*. Paper presented at the American Association for Artificial Intelligence 2005 Educational Data Mining Workshop.

Barnes, T., Stamper, J. & Madhyastha, T. (2006). *Comparative Analysis of Concept Derivation Using the Q-matrix Method and Facets*.

Baschera, G.-M., Busetto, A., Klingler, S., Buhmann, J. & Gross, M. (2011). Modeling Engagement Dynamics in Spelling Learning. In G. Biswas, S. Bull, J. Kay & A. Mitrovic (Eds.), *Artificial Intelligence in Education* (Vol. 6738, pp. 31-38): Springer Berlin Heidelberg.

Beck, J. & Mostow, J. (2008). How Who Should Practice: Using Learning Decomposition to Evaluate the Efficacy of Different Types of Practice for Different Types of Students. In (pp. 353-362).

Brown, J. S. & VanLehn, K. (1980). Repair theory: A generative theory of bugs in procedural skills. *Cognitive Science*, *4*, 379-426.

Burton, R. (1982). Diagnosing bugs in a simple procedural skill. In D. Sleeman & J. Brown (Eds.), *Intelligent Tutoring Systems* (pp. 157-184): Academic Press.

Cai, Z., Graesser, A. C., Forsyth, C. M., Burkett, C., Millis, K., Wallace, P., et al. (2011). Trialog in ARIES: User input assessment in an intelligent tutoring system. In W. C. S. Li (Ed.), *Proceedings of the 3rd IEEE International Conference on Intelligent Computing and Intelligent Systems* (pp. 429–433). Guangzhou, P.R. China: IEEE Press.

Calfee, R. C. & Atkinson, R. C. (1965). Paired-associate models and the effects of list length. *Journal of Mathematical Psychology, 2*, 254-265.

Carrier, M. & Pashler, H. (1992). The influence of retrieval on retention. *Memory & Cognition, 20*, 633-642.

Cen, H., Koedinger, K. R. & Junker, B. (2006). Learning Factors Analysis - A general method for cognitive model evaluation and improvement. In *Proceedings of the 8th International Conference on Intelligent Tutoring Systems* (pp. 164-175): Springer Berlin / Heidelberg.

Cen, H., Koedinger, K. R. & Junker, B. (2007). *Is Over Practice Necessary? – Improving Learning Efficiency with the Cognitive Tutor through Educational Data Mining*. Paper presented at the 13th International Conference on Artificial Intelligence in Education, Los Angeles, CA.

Cen, H., Koedinger, K. R. & Junker, B. (2008). *Comparing two IRT models for conjunctive skills*. Paper presented at the Proceedings of the 9th International Conference on Intelligent Tutoring Systems, Montreal, Canada.

Cerri, S. A., Clancey, W. J., Papadourakis, G. & Panourgia, K. (2012). *Intelligent Tutoring Systems - 11th International Conference, ITS 2012, Chania, Crete, Greece, June 14-18, 2012. Proceedings* (Vol. 7315): Springer.

Chi, M., Koedinger, K. R., Gordon, G., Jordan, P. & VanLehn, K. (2011). *Instructional Factors Analysis: A Cognitive Model For Multiple Instructional Interventions*. Poster presented at the 4th International Conference on Educational Data Mining, Eindhoven, The Netherlands.

Chi, M. T. H. (2009). Active-Constructive-Interactive: A Conceptual Framework for Differentiating Learning Activities. *Topics in Cognitive Science, 1*, 73-105.

Chi, M. T. H., Siler, S. A., Jeong, H., Yamauchi, T. & Hausmann, R. G. (2001). Learning from human tutoring. *Cognitive Science, 25*, 471-533.

Corbett, A. T. & Anderson, J. R. (1992). Student modeling and mastery learning in a computer-based programming tutor. In C. Frasson, G. Gauthier & G. McCalla (Eds.), *Intelligent Tutoring Systems: Second International Conference on Intelligent Tutoring Systems* (pp. 413-420). New York: Springer-Verlag.

Corbett, A. T. & Anderson, J. R. (1995). Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Modeling and User-Adapted Interaction, 4*, 253–278.

Crowder, N. A. (1959). Automatic tutoring by means of intrinsic programming. In E. Galanter (Ed.), *Automatic teaching: The state of the art* (pp. 109-116). New York: Wiley & Sons.

D'Mello, S. K. & Graesser, A. (2010). Multimodal semi-automated affect detection from conversational cues, gross body language, and facial features. *User Modeling and User-Adapted Interaction, 20*, 147-187.

D'Mello, S. K., Graesser, A. & King, B. (2010). Toward Spoken Human-Computer Tutorial Dialogues. *Human Computer Interaction, 25*, 289-323.

D'Mello, S. K., Olney, A. M. & Person, N. (2010). Mining Collaborative Patterns in Tutorial Dialogues. *Journal of Educational Data Mining, 2*, 1-37.

D'Mello, S. K. & Graesser, A. C. (in press). AutoTutor and affective AutoTutor: Learning by talking with cognitively and emotionally intelligent computers that talk back. *ACM Transactions on Interactive Intelligent Systems*.

Desmarais, M. C. & Baker, R. S. J. d. (2012). A review of recent advances in learner and skill modeling in intelligent learning environments. *User Modeling and User-Adapted Interaction, 22*, 9-38.

Desmarais, M. C., Maluf, A. & Liu, J. (1996). User-expertise modeling with empirically derived probabilistic implication networks. *User Modeling and User-Adapted Interaction, 5*, 283-315.

Desmarais, M. C. & Pu, X. (2005). A Bayesian Student Model without Hidden Nodes and its Comparison with Item Response Theory. *Int. J. Artif. Intell. Ed., 15*, 291-323.

Doignon, J.-P. & Falmagne, J.-C. (1999). *Knowledge spaces*: Springer.

Draney, K. L., Pirolli, P. & Wilson, M. (1995). A measurement model for a complex cognitive skill. In P. D. Nichols, S. F. Chipman & R. L. Brennan (Eds.), *Cognitively diagnostic assessment* (pp. 103–125).

Elsom-Cook, M. (1993). Student modelling in intelligent tutoring systems. *Artificial Intelligence Review, 7*, 227-240.

Falmagne, J.-C., Doignon, J.-P., Cosyn, E. & Thiery, N. (2003). The assessment of knowledge in theory and in practice. *Institute for Mathematical Behavioral Sciences, Paper 26*.

Fox, B. A. (1993). *The human tutoring dialogue project*. Hillsdale, NJ: Lawrence Erlbaum Associates, Inc.

Goldberg, B., Sottilare, R., Brawner, K. & Holden, H. (2011). Predicting Learner Engagement during Well-Defined and Ill-Defined Computer-Based Intercultural Interactions. In S. D'Mello, A. Graesser, B. Schuller & J.-C. Martin (Eds.), *Affective Computing and Intelligent Interaction* (Vol. 6974, pp. 538-547): Springer Berlin Heidelberg.

Gong, Y., Beck, J. & Heffernan, N. T. (2010). Comparing Knowledge Tracing and Performance Factor Analysis by Using Multiple Model Fitting Procedures. In V. Aleven, J. Kay & J. Mostow (Eds.), *Intelligent Tutoring Systems* (Vol. 6094, pp. 35-44): Springer Berlin / Heidelberg.

Graesser, A. C., Conley, M. W. & Olney, A. (2012). Intelligent tutoring systems. In K. R. H. telligent tutoring systems, S. Graham, T. Urdan, A. G. Bus, S. Major & H. L. Swanson (Eds.), *APA educational psychology handbook, Vol 3: Application to learning and teaching* (pp. 451-473). Washington, DC, US: American Psychological Association.

Graesser, A. C., D'Mello, S. K., Xiangen, H., Cai, Z., Olney, A. & Morgan, B. (2012). AutoTutor. *Applied Natural Language Processing: Identification, Investigation, and Resolution.*, pp. 169-187.

Graesser, A. C., Penumatsa, P., Ventura, M., Cai, Z. & Hu, X. (2007). Using LSA in AutoTutor: Learning through mixed initiative dialogue in natural language. In D. M. T. Landauer, S. Dennis & W. Kintsch (Ed.), *Handbook of latent semantic analysis* (pp. 234-262). Mahwah, NJ: Erlbaum.

Graesser, A. C. & Person, N. K. (1994). Question Asking during Tutoring. *American Educational Research Journal, 31*, 104-137.

Graesser, A. C., Person, N. K. & Magliano, J. P. (1995). Collaborative dialogue patterns in naturalistic one-to-one tutoring. *Applied Cognitive Psychology, 9*, 1-28.

Groen, G. J. & Atkinson, R. C. (1966). Models for optimizing the learning process. *Psychological Bulletin, 66*, 309-320.

Heffernan, N. T., Koedinger, K. R. & Razzaq, L. (2008). Expanding the Model-Tracing Architecture: A 3rd Generation Intelligent Tutor for Algebra Symbolization. *International Journal of Artificial Intelligence in Education, 18*, 153-178.

Heffernan, N. T., Turner, T. E., Lourenco, A. L. N., Macasek, M. A. & Nuzzo-Jones, G. The ASSISTment builder: Towards an analysis of cost effectiveness of ITS creation.

Hu, X., Craig, S. D., Bargagliotti, A. E., Graesser, A. C., Okwumabua, T., Anderson, C., et al. (2012). The Effects of a Traditional and Technology-based After-school Setting on 6th Grade Student's Mathematics Skills. *Journal of Computers in Mathematics and Science Teaching, 31*, 17-38.

Johnson, W. L. & Valente, A. (2008). Tactical language and culture training systems: using artificial intelligence to teach foreign languages and cultures, *Proceedings of the 20th national conference on Innovative applications of artificial intelligence - Volume 3* (pp. 1632-1639). Chicago, Illinois: AAAI Press.

Kapoor, A. & Picard, R. W. (2005). *Multimodal affect recognition in learning environments*. Paper presented at the Proceedings of the 13th annual ACM international conference on Multimedia, Hilton, Singapore.

Karpicke, J. D. & Roediger, H. L., III. (2008). The critical importance of retrieval for learning. *Science, 319*, 966–968.

Koedinger, K. R., Pavlik Jr., P. I., Stamper, J., Nixon, T. & Ritter, S. (2011). Fair blame assignment in student modeling. In M. Pechenizkiy, T. Calders, C. Conati, S. Ventura, C. Romero & J. Stamper (Eds.), *Proceedings of the 4th International Conference on Educational Data Mining* (pp. 91–100). Eindhoven, the Netherlands.

Lahti, L. (2010). *Personalized learning paths based on Wikipedia article statistics*. Paper presented at the CSEDU 2010.

Landauer, T. K., McNamara, D. S., Dennis, S. E. & Kintsch, W. E. (2007). *Handbook of latent semantic analysis*: Lawrence Erlbaum Associates Publishers.

Lepper, M. R. & Hodell, M. (1989). Intrinsic Motivation in the Classroom. *Research on Motivation in Education: Goals and cognitions, 3*, 73.

Litman, D. (2013). Speech and language processing for adaptive training. In P. Durlach & A. Lesgold (Eds.), *Adaptive technologies for training and education.*: Cambridge University Press.

Lumsdaine, A. A. & Glaser, R. E. (1960). *Teaching Machines and Programmed Learning, a Source Book*. Washington DC. : National Education Association, Dept. of Audiovisual Instruction.

Mayo, M. & Mitrovic, A. (2001). Optimising ITS Behaviour with Bayesian Networks and Decision Theory. *International Journal on Artificial Intelligence in Education, 12*, 124-153.

Mayo, M., Mitrovic, A. & McKenzie, J. (2000). CAPIT: An Intelligent Tutoring System for Capitalisation and Punctuation, *International Workshop on Advanced Learning Technologies* (Vol. 0, pp. 151-154). Palmerston North, New Zealand: IEEE Computer Society.

McTear, M. F. (2002). Spoken dialogue technology: enabling the conversational user interface. *ACM Computing Surveys (CSUR), 34*, 90-169.

Millis, K., Forsyth, C., Butler, H., Wallace, P., Graesser, A. & Halpern, D. (2011). Operation ARIES!: A Serious Game for Teaching Scientific Inquiry. *Serious Games and Edutainment Applications,* pp. 169-195.

Mitrovic, A. (1998). Experiences in Implementing Constraint-Based Modeling in SQL-Tutor. In B. Goettl, H. Halff, C. Redfield & V. Shute (Eds.), *Intelligent Tutoring Systems* (Vol. 1452, pp. 414-423): Springer Berlin Heidelberg.

Mitrovic, A. (2003). An Intelligent SQL Tutor on the Web. *International Journal of Artificial Intelligence in Education, 13*, 173-197.

Mitrovic, A. (2012). Fifteen years of constraint-based tutors: what we have achieved and where we are going. *User Modeling and User-Adapted Interaction, 22*, 39-72.

Mitrovic, A., Koedinger, K. R. & Martin, B. (2003). A Comparative Analysis of Cognitive Tutoring and Constraint-Based Modeling. In *User Modeling 2003* (Vol. 2702/2003): Springer Berlin / Heidelberg.

Mitrovic, A. & Martin, B. (2007). Evaluating the Effect of Open Student Models on Self-Assessment. *International Journal of Artificial Intelligence in Education, 17,* 121-144.

Mitrovic, A., Martin, B. & Suraweera, P. (2007). Intelligent Tutors for All: The Constraint-Based Approach. *IEEE Intelligent Systems, 22*, 38-45.

Mitrovic, A., Martin, B., Suraweera, P., Zakharov, K., Milik, N., Holland, J., et al. (2009). ASPIRE: An Authoring System and Deployment Environment for Constraint-Based Tutors. *Int. J. Artif. Intell. Ed., 19,* 155-188.

Mitrovic, A. & Ohlsson, S. (1999). Evaluation of a Constraint-Based Tutor for a Database. *International Journal of Artificial Intelligence in Education, 10*, 238-256.

Mitrovic, A., Ohlsson, S. & Barrow, D. K. (2013). The effect of positive feedback in a constraint-based intelligent tutoring system. *Computers & Education, 60*, 264-272.

Mitrovic, A. & Weerasinghe, A. (2009). Revisiting the Ill-Definedness and Consequences for ITSs. In V. Dimitrova, R. Mizoguchi, B. du Boulay & A. Graesser (Eds.), *Proc 14th Int Conf AIED* (pp. 375-382).

Moore, J. D. (1994). *Participating in explanatory dialogues: interpreting and responding to questions in context*. Cambridge, MA, USA: MIT Press.

Nkambou, R., Mizoguchi, R. & Bourdeau, J. (2010). *Advances in Intelligent Tutoring Systems* (Vol. 308): Springer.

O'Reilly, R. C. (1998). Six principles for biologically based computational models of cortical cognition. *Trends in Cognitive Sciences*, *2*, 455-462.

Ohlsson, S. (1992). Constraint-based student modelling. *International Journal of Artificial Intelligence in Education, 3*, 429-447.

Ohlsson, S. & Mitrovic, A. (2007). Fidelity and Efficiency of Knowledge Representations for Intelligent Tutoring Systems. *Technology, Instruction, Cognition and Learning (TICL), 5*, 101-132.

Olney, A. M. (2009). GnuTutor: An open source intelligent tutoring system, *Proceedings of the 14th International Conference on Artificial Intelligence in Education* (pp. 803). Brighton UK: Amsterdam: IOS Press.

Olney, A. M., Louwerse, M., Mathews, E., Marineau, J., Hite-Mitchell, H. & Graesser, A. C. (2003). Utterance Classification in AutoTutor, *Proceedings of the HLT-NAACL 03 Workshop on Building Educational Applications Using Natural Language Processing* (pp. 1-8). Philadelphia: Association for Computational Linguistics.

Olney, A. M., Person, N. K. & Graesser, A. C. (2012). Guru: Designing a Conversational Expert Intelligent Tutoring System. *Cross-Disciplinary Advances in Applied Natural Language Processing: Issues and Approaches*, pp. 156-171.

Pavlik Jr., P. I. (2007). Understanding and applying the dynamics of test practice and study practice. *Instructional Science*, *35*, 407–441.

Pavlik Jr., P. I. & Anderson, J. R. (2008). Using a model to compute the optimal schedule of practice. *Journal of Experimental Psychology: Applied, 14*, 101–117.

Pavlik Jr., P. I., Cen, H. & Koedinger, K. R. (2009). Performance factors analysis -- A new alternative to knowledge tracing. In V. Dimitrova, R. Mizoguchi, B. d. Boulay & A. Graesser (Eds.), *Proceedings of the 14th International Conference on Artificial Intelligence in Education* (pp. 531–538). Brighton, England.

Pavlik Jr., P. I., Presson, N., Dozzi, G., Wu, S.-m., MacWhinney, B. & Koedinger, K. R. (2007). The FaCT (Fact and Concept Training) System: A new tool linking cognitive science with educators. In D. McNamara & G. Trafton (Eds.), *Proceedings of the Twenty-Ninth Annual Conference of the Cognitive Science Society* (pp. 1379–1384). Mahwah, NJ: Lawrence Erlbaum.

Pavlik Jr., P. I. & Toth, J. (2010). How to build bridges between intelligent tutoring system subfields of research. In J. Kay, V. Aleven & J. Mostow (Eds.), *Proceedings of the 10th International Conference on Intelligent Tutoring Systems, Part II* (pp. 103–112). Pittsburgh, PA: Springer.

Pavlik Jr., P. I., Yudelson, M. & Koedinger, K. R. (2011). Using contextual factors analysis to explain transfer of least common multiple skills. In G. Biswas, S. Bull, J. Kay & A. Mitrovic (Eds.), *Artificial Intelligence in Education* (Vol. 6738, pp. 256–263). Berlin, Germany: Springer.

Person, N. K. & Graesser, A. C. (2003). Fourteen facts about human tutoring: Food for thought for ITS developers, *AI-ED 2003 Workshop Proceedings on Tutorial Dialogue Systems: With a View Toward the Classroom* (pp. 335-344).

Person, N. K., Graesser, A. C., Magliano, J. P. & Kreuz, R. J. (1994). Inferring what the student knows in one-to-one tutoring: The role of student questions and answers. *Learning and Individual Differences, 6*, 205–229.

Peterson, L. R. (1965). Paired-associate latencies after the last error. *Psychonomic Science, 2,* 167-168.

Psotka, J., Massey, L. D. & Mutter, S. A. (1988). *Intelligent tutoring systems: Lessons learned*: Lawrence Erlbaum.

Rey-López, M., Fernández-Vilas, A., Díaz-Redondo, R., Pazos-Arias, J. & Bermejo-Muõz, J. (2006). Extending SCORM to Create Adaptive Courses. In W. Nejdl & K. Tochtermann (Eds.), *Innovative Approaches for Learning and Knowledge Sharing* (Vol. 4227, pp. 679-684): Springer Berlin Heidelberg.

Rickard, T. C. (1997). Bending the power law: A CMPL theory of strategy shifts and the automatization of cognitive skills. *Journal of Experimental Psychology: General, 126*, 288-311.

Rickard, T. C. (1999). A CMPL alternative account of practice effects in numerosity judgment tasks. *Journal of Experimental Psychology: Learning, Memory & Cognition, 25*, 532-542.

Ritter, S. & Anderson, J. (2006). *Cognitive Tutor: Tracking learning in real time. Testimony to the National Mathematics Panel*.

Ritter, S., Anderson, J. R., Koedinger, K. R. & Corbett, A. (2007). Cognitive Tutor: Applied research in mathematics education. *Psychonomic Bulletin & Review, 14*, 249-255.

Rus, V. & Graesser, A. C. (2009). *The Question Generation Shared Task and Evaluation Challenge*: University of Memphis.

Rus, V., McCarthy, P., Graesser, A. & McNamara, D. (2009). Identification of Sentence-to-Sentence Relations Using a Textual Entailer. *Research on Language and Computation, 7*, 209-229.

Schunn, C. D. (2005). Evaluating goodness-of-fit in comparison of models to data. In W. Tack (Ed.), *Psychologie der Kognition: Reden and Vorträge anlässlich der Emeritierung von Werner Tack* (pp. 115-154). Saarbrueken, Germany: University of Saarland Press.

Segalowitz, N. S. & Segalowitz, S. J. (1993). Skilled performance, practice, and the differentiation of speed-up from automatization effects - evidence from 2nd-language word recognition. *Applied Psycholinguistics, 14*, 369-385.

Shah, F., Evens, M., Michael, J. & Rovick, A. (2002). Classifying Student Initiatives and Tutor Responses in Human Keyboard-to-Keyboard Tutoring Sessions. *Discourse Processes, 33*, 23-52.

Sleeman, D. & Brown, J. S. (1982). *Intelligent tutoring systems*. New York: Academic Press.

Sleeman, D., Ward, R. D., Kelly, E., Martinak, R. & Moore, J. (1991). An overview of recent studies with PIXIE. In P. Goodyear (Ed.), *Teaching Knowledge and Intellgient Tutoring* (pp. 173-185).

Smallwood, R. D. (1962). *A decision structure for teaching machines*. Cambridge: MIT Press.

Smallwood, R. D. (1971). The analysis of economic teaching strategies for a simple learning model. *Journal of Mathematical Psychology, 8*, 285-301.

Spada, H. & McGraw, B. (1985). The assessment of learning effects with linear logistic test models. In S. Embretson (Ed.), *Test design: Developments in psycholgoy and psychometrics*. Orlando, FL: Academic Press.

Suraweera, P. & Mitrovic, A. (2004). An Intelligent Tutoring System for Entity Relationship Modelling. *International Journal of Artificial Intelligence in Education, 14*, 375-417.

Tatsuoka, K. K. (1983). Rule space: An approach for dealing with misconceptions based on item response theory. *Journal of Educational Measurement, 20*, 345-354.

Thomas, C., Davies, I., Openshaw, D. & Bird, J. (1963). *Programmed Learning in Perspective: A Guide to Program Writing*: Aldine De Gruyter.

Thompson, C. P., Wenger, S. K. & Bartling, C. A. (1978). How recall facilitates subsequent recall: A reappraisal. *Journal of Experimental Psychology: Human Learning & Memory, 4*, 210-221.

VanLehn, K. (2006). The behavior of tutoring systems. *International Journal of Artificial Intelligence in Education, 16*, 227-265.

VanLehn, K., Graesser, A. C., Jackson, G. T., Jordan, P., Olney, A. M. & Rosé, C. P. (2007). When are tutorial dialogues more effective than reading? *Cognitive Science, 31*, 3–62.

VanLehn, K., Jones, R. M. & Chi, M. T. H. (1992). A Model of the Self-Explanation Effect. *Journal of the Learning Sciences*, *2*, 1 - 59.

Weerasinghe, A., Mitrovic, A. & Martin, B. (2009). Towards Individualized Dialogue Support for Ill-Defined Domains. *International Journal of Artificial Intelligence in Education, 19*, 357-379.

Wixon, M., Baker, R. S. J. d., Gobert, J. D., Ocumpaugh, J. & Bachmann, M. (2012). *WTF? detecting students who are conducting inquiry without thinking fastidiously*. Paper presented at the Proceedings of the 20th international conference on User Modeling, Adaptation, and Personalization, Montreal, Canada.

Woolf, B. P. (2008). *Building intelligent interactive tutors: Student-centered strategies for revolutionizing e-learning*. San Francisco, CA: Morgan Kaufmann.

Woolf, B. P. (2010). *A roadmap for education technology*: National Science Foundation.

Zakharov, K., Mitrovic, A. & Ohlsson, S. (2005). *Feedback Micro-engineering in EER-Tutor*. Paper presented at the Proceeding of the 2005 conference on Artificial Intelligence in Education: Supporting Learning through Intelligent and Socially Informed Technology.

Section II

Current Learner Modeling Tools and Methods

H. Holden, Ed.

CHAPTER 6 –Understanding Current Learner Modeling Approaches

Heather K. Holden
U.S. Army Research Laboratory (ARL) - Human Research and Engineering Directorate (HRED)

Introduction

We've learned in the previous section some of the fundamental limitations and challenges that surround learner modeling development. However, this section continues to provide insight on these challenges and design recommendations for GIFT based on a current perspective of learner modeling tools and methods. As the core module of an ITS, the learner model is a representation of the learner's state of knowledge at any given time. Ideally, this model would be comprehensive enough to include and analyze information on the learner's individual difference characteristics as well as past, current, and predicted competencies, performance, cognition, affect, behaviors, etc. This model would also be flexible enough to support a variety of learning/instructional activities and types of learners. Realistically, such a model does not exist among current learner modeling research.

Current State of Learner Modeling Research

Based on the previous section of this book, we observe that learner models are built for different purposes, such as recognizing solutions paths, evaluating problem-solving abilities, or describing constraints for violations made by the learner. Current techniques of generating learner models include Bayesian networks, belief networks, case-based reasoning, and expectation maximization. Methods such as model-tracing are more cost-effective, but do not have the ability to record or monitor learner's behavior. Learner models are commonly classified according to their relationship to an expert model, but can be classified by their performing function (i.e., corrective, elaborative, strategic, diagnostic, predictive, or evaluative).

The content within learner models is usually categorized in two components: domain-specific or domain-independent information (learner-specific characteristics/individual differences). Domain-specific information represents a reflection of the learner's state and level of knowledge or ability within a particular domain. This type of information primarily includes historical competency (domain knowledge and skills measured over time), misconceptions, problem solving strategies, etc. Domain-independent information consists of all relevant characteristics of an individual learner and can include, but is not limited to, the following elements: learning goals; cognitive aptitudes; measures of motivational state; learning preferences (including styles and personality); interest; demographics; past performance and competency (non-domain-specific); behavioral/psychological measures; cognitive and affective dimensions; and personal control beliefs (including general self-efficacy; locus of control).

First-generation ITS implementations primarily adapted instruction based on learner performance and current state of knowledge domain-specific information. These systems used learner models with corrective or elaborative functionality, but lacked any strategic, diagnostic, or predictive capabilities. The advantage of modeling this type of information is that it allows the model to be more generalized across multiple populations. Although useful, such information alone is not sufficient enough for providing the highly adaptive individualized training needed for ITSs of the next generation. Learner characteristics can be significantly different between learners and, collectively, is not the same for any two learners.

Primary sub-research areas of current learner modeling include, but are not limited to, learner state classifications, cognitive modeling, affect modeling, the impact of individual differences, behavioral and physiological sensing, and performance assessments. As the demand for higher adaptation and flexibility of learner models increases, so does the necessity to understand the interrelationship between all aspects of learner modeling content and assessment accuracy. Within the past 10 years, learner model research has extended to consider a broader range of learner characteristics as difficulty in addressing learner's knowledge gaps has become more apparent. Since the beginning, the learner modeling research community continues to help address this two-part question: *what aspects of the learner should be modeled and how can we achieve the best possible levels of state and performance classification and predictive accuracy?* Therefore, we can observe an increase in studies blending the sub-areas of learner modeling research as well as the emergence of other key areas of student modeling research, i.e., motivation, disengagement, metacognition, self-regulated learning, open learner modeling, group and collaborative learner modeling, and long-term learner modeling.

These newer areas of research present their own challenges in addition to the ones already surrounding the general research area. Some of these areas are covered in a later section of this book; however, the purpose of this section is to present the "bridge" to a new era of learner modeling research. The five chapters of this section discuss current areas and challenges surrounding learner modeling. Each chapter presents its own voice to a particular area of interest and gives recommendations on how GIFT can use the information within the chapter to enhance future GIFT versions.

The chapter by Tomar and Nielsen, entitled *Affective-Behavioral-Cognitve (ABC) Learner Modeling*, presents a framework for modeling interlocutors (i.e., users of e-learning systems) that integrates inductive and abductive reasoning over observations including the interlocutors' past and current behavior to develop a joint model for predicting their emotions, behaviors, and cognitive states. Their Affective-Behavioral-Cognitive (ABC) Learner Model follows an approach that users' behavioral responses can be a path to predict, recognize, and interpret their affective state. These behavioral responses are analyzed using a cognitive theory of emotions, which gives us inferences about the possible affective states of the learner. An appraisal component of the model relies on the desirability of events based on the learners' objectives, the affective and cognitive states predicted to result from the events, and the consequent expected behaviors. They discuss the specific aspects of their model and the interrelationship between these elements; provide an example of how the model can perform within an e-learning scenario; and highlight recommendations for how GIFT can use such a model for its learner modeling approaches.

The chapter by Holden and Sinatra, entitled *The Need for Empirical Evaluation of Learner Model Elements*, highlight issues with the lack of standardization on the structure of learner models and modeling techniques as it inhibits the validation and reusability of learner model elements. They argue that there are several aspects of user modeling, of which learner modeling is a subset, which have yet to be explored by the ITS learner modeling community. Such factors include learner's expertise, skills, attitudes, perceptions, and self-efficacy toward both computers/technology in general and the specific ITS. As we progress toward extending ITSs to be inclusive of job-related training and education, these factors may prove important in classificating learners' states, performance, and system behaviors. They also argue that more empirical evaluations are essential to better understanding the impact and interaction effects of current and potential learner model elements. Practical implications for researchers as well as design recommendations for such evaluations in GIFT are provided within this chapter.

The chapter by Hu, Morrison, and Cai, entitled *On the Use of Learner Micromodels as Partial Solutions to Complex Problems in a Multi-agent, Conversation-based Intelligent Tutoring System*, argues that at some point a general-purpose system, like GIFT, would employ an open, multi-agent architecture in which some agents will perform simple tasks, while others will take on more complex ones, such as generating appropriate responses to user questions. They describe an autonomous software agent that produces turn-by-turn analysis of a user's discourse moves on two dimensions: *novelty* and *relevance*. This process includes building a "*micromodel*" of the learner's current state, including a relevance-novelty measure for single turns and a series of turns. They provide an example of a highly specialized agent within a currently existing multi-agent, conversation-based ITS architecture that is capable of making assertions about the learner micromodel information that is of value to other agents and/or for their own purposes.

The chapter by Douglass, entitled *Learner Models in the Large-Scale Cognitive Modeling Initiative*, presents an Air Force Research Laboratory (AFRL) research effort to develop training capabilities for live, virtual, and constructive systems, as such system face challenges of (1) increasing the scale of cognitive models and (2) integrating them into software-intensive training environments. The AFRL Large-Scale Cognitive Modeling (LSCM) initiative is developing solutions to these scale and interoperability challenges based on high-level languages for describing cognitive models and net-centric simulation frameworks supporting them. This chapter introduces the LSCM initiative and explains how learner models are represented and used in the systems developing in its scope. The author illustrates how formal models of behavior models are specified and used to track events and build/refine representations of the knowledge, conceptual weaknesses, and procedural skills of monitored learners. He also explains how performance prediction and optimization capabilities based on mathematical extensions of the General Performance Equation (Anderson & Schunn, 2000) monitor learner actions, trace models of behavior, and use knowledge about learners to track and predict performance.

After reading these chapters, we can see that the lack of consensus and standardization for developing learner models is still apparent just as in the previous book section. We are now at the point of which such research can no longer ignore the necessity of building and enhancing standards for creating learner models with higher-level functionality to fulfill the *ideal* vision previously mentioned. Each of the chapters within this section provides a suggested method for aiding GIFT's future learner model design and functionality. Tomar and Nielsen, emphasize the importance of assessing the interplay between learner's cognition, affect, and behavior. Holden and Sinatra, emphasize the possibility of GIFT's ability to do comparison between learner model elements to support the validation of impact and interactions between learner characteristics. The last two chapters provide explicit examples and implementations of work-in-progress. Hu, Morrison, and Cai propose the use of multi-agents and micromodels as advantageous for reusability, a key element and motivation of GIFT. Moreover, Douglass highlights similarities between AFRL's LSCM and ARL's GIFT and views the use of research modeling language (RML) intelligent agents to effectively use behavior models to trace trainee's actions as beneficial to include within GIFT's design.

Recommendations for GIFT's Immediate Direction

Each of the chapters within this section provided recommendations for GIFT in regards to its learner modeling approaches. In sum, they are as follows:

1. GIFT should consider incorporating the ABC model to observe learner performance over a period of time and to create affective and cognitive profiles which have threshold values and decay rates associated with the states in consideration. This process will allow GIFT's learner model to determine states more effectively.

2. Future learner modeling researchers should consider using GIFT for their research since the system provides plans to have the ability to interchange and learner models and its elements. Therefore, researchers will be able to understand how learner-specific characteristics,

perceptions, and preferences interplay and influence the learning process as well as how they dynamically change throughout instruction.

3. GIFT should consider adopting a common agent communication language (ACL) as the bases for a new generation of agent-based intelligent learning systems that are capable of autonomous cooperation. The instantiation of a common ACL and the development of a shared ontology can bring great benefits to GIFT's ultimate vision.

4. Integration of AFRL's LSCM/RML and ARL's GIFT would be an advantageous system that could serve as a technical solution to the problem of monitoring trainee actions and delivering contextually relevant instruction.

References

Anderson, J.R. & Schunn, C. D. (2000). Implications of the ACT-R learning theory: No magic bullets. In R. Glaser (Ed.), *Advances in instructional psychology: Educational design and cognitive science (Vol. 5),* (pp. 1-34). Mahwah, NJ: Erlbaum.

CHAPTER 7 –Affective-Behavioral-Cognitive (ABC) Learner Modeling

Abhiraj Tomar and Rodney D. Nielsen
University of North Texas

Introduction

This chapter presents a framework for modeling users of e-learning systems that integrates inductive and abductive reasoning over observations including the learner's past and current behavior to develop a joint model for predicting emotions, behaviors, and cognitive states. This ABC Learner Model follows an approach that learners' behavioral responses can be a path to predict, recognize, and interpret their affective state. These behavioral responses are analyzed using a cognitive theory of emotions, which gives us inferences about the possible affective states of the learner. An appraisal component of the model relies on the desirability of events given the learner's objectives, the resulting affective and cognitive states predicted to result from the events, and the consequent behaviors expected.

Most current techniques for modeling learners make relatively strong assumptions about what affective, behavioral, and cognitive states are best for learning, and about what to do in a single interaction turn to maximize immediate learning based on the current states. Most ITSs typically do not learn which user states optimize short- or long-term learning goals, learn what sequence of events will likely elicit particular user states, appraise system attempts toward these goals, or learn appropriate corrective actions based on this appraisal. The ABC Model attempts to fill these gaps by learning to make predictions that take these more complex relations between events, learner states, and time-dependent scenarios into account.

We begin by describing user models in general and the various user states incorporated by the ABC Model. We explain a cognitive theory of affect and behavior and its use in creating user profiles. In the following section, we give an overview of the various categories of knowledge and information about the learner, which are required for creating a learner-specific profile and the methods for obtaining them. After this, we discuss the categorization of events that can occur in an e-learning scenario and the relationship these events have with the learner models. Then, we move on to describe the ABC Model's appraisal mechanism, which relates learner states to their goals. Based on this, the system provides suggestions for the learner in support of the e-learning process. Finally, the system updates the learner model based on observations of the effects resulting from its decisions. In the last section, we provide recommendations for GIFT and how the various aspects of the ABC Model can support the modules present in GIFT to enhance their performance for e-learning environments.

The ABC User Model

The ABC User Model (Figure 7-1) tracks the affective, behavioral, and cognitive states and patterns of the user and applies a cognitive theory of emotions to infer and analyze these states and patterns. This analysis is then used to provide an adaptive and interactive e-learning environment to learners intended to optimize learning based on their individual characteristics, requirements, and preferences. The *affective states* store the information relevant to the learner's emotions. The *behavioral patterns* store the information associated with the way the learner interacts with and reacts to events within the system and their apparent objectives. The *cognitive states* store information associated with the learner's mental processes. This section gives a description of these states and the aspects covered by each.

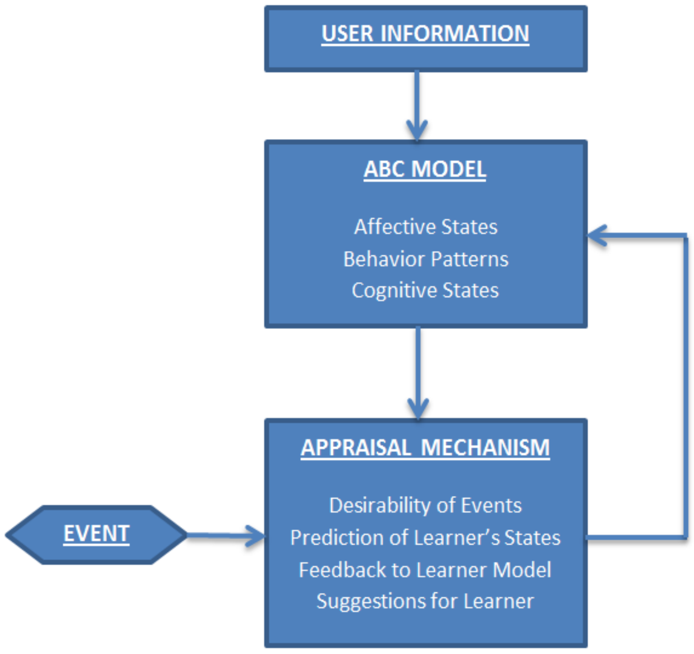


Figure 7-1. The ABC Model

Affective States

The learner model needs to represent and acquire the learner's emotional states. It needs to develop techniques to make predictions and inferences about and based on these emotions. In the ABC Model, emotions are ascribed to the learner based on not only the sensory inputs, but also the learner's behavior and on the events of the world (Martinho et al., 2000). Hence, it needs a cognitive theory of emotions that considers and works with such stimuli. Ortony, Clore, and Collins' Theory of Emotions, OCC Theory (Ortony et al., 1988), is one such cognitive appraisal theory and forms the basis for the affective aspect of the ABC Model.

OCC Theory: Ortony, Clore, and Collins proposed a cognitive appraisal theory that is structured as a three-branch typology, corresponding to three kinds of stimuli: consequences of events, actions of agents, and aspects of objects. Each kind of stimulus is appraised with respect to one central criterion, called the central appraisal variable (Adam et al., 2009). An individual judges the following:

1. The desirability of an event
2. The approbation of an action
3. The attraction of an object

The OCC typology contains 22 emotions, grouped in 6 classes. These are depicted in Figure 7-2. The first branch contains only one class of emotions related to aspects of objects, triggered by the appraisal of the objects with respect to the individual's likings. This class is seldom involved with e-learning environments, usually limited to the cases having virtual characters involved with the learning process. The second branch contains three classes of emotions triggered by the appraisal of the consequences of an event as to its desirability. Different emotions arise depending on the prospect and the focus of the desirability of consequences of events. Individuals can focus desirability either on themselves or on other individuals. The third branch contains two classes of emotions triggered by the *actions of agents*, which are appraised according to their compliance and conformity to norms and standards.

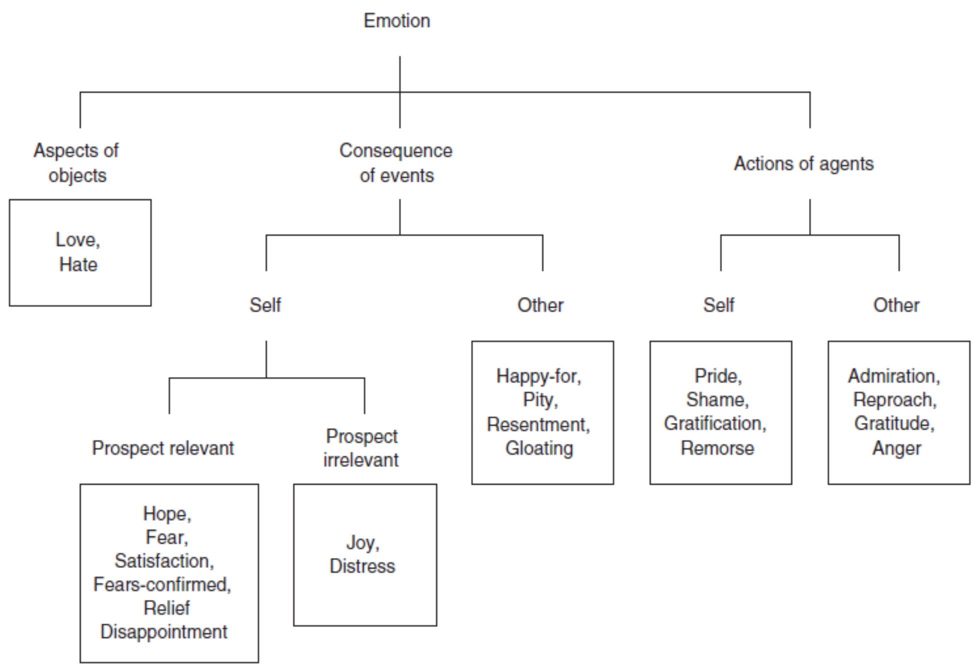

Emotion
Aspects of objects
Consequence of events
Actions of agents
Love, Hate
Self
Other
Self
Other
Prospect relevant
Prospect irrelevant
Happy-for, Pity, Resentment, Gloating
Pride, Shame, Gratification, Remorse
Admiration, Reproach, Gratitude, Anger
Hope, Fear, Satisfaction, Fears-confirmed, Relief Disappointment
Joy, Distress

Figure 7-2. The OCC Theory of Emotions typology

Based on the OCC theory, the learner's affective states can be characterized by two types of data: the learner's *emotional profile* and the *emotional states* (Martinho et al., 2000).

Emotional Profile: The emotional profile represents the emotional pattern exhibited by the learner and is constituted by the following:

- *Emotional Class Thresholds*: These represent the assumed emotional "resistance" to the different classes of emotions.
    - For instance, emotional thresholds can model how easily the learner can be disappointed upon not being able to solve a problem. The amount of effort produced by a learner in trying to solve a difficult problem can be used to state the learner's level of resistance to disappointment. If learners have a tendency of not producing enough effort in solving problems related to topics in which they have been performing poorly, it indicates a low level of resistance to disappointment.

- *Emotional Class Decays*: These represent the extent to which emotions represent self-sustaining processes or how long the emotions being experienced by the learner last.
    - It is very difficult to assign emotional decay values, but they can have a significant effect on how the learner behaves in a learning session. They can be inferred from past trends, for instance, some learners get excited upon solving a problem they perceive as difficult and, in such a state, they might make mistakes while working on subsequent relatively easy problems. A decay rate for such an emotion can be assigned by observing patterns and the average time required for saturation.

Emotional States: Two types of emotional states are stored in an ABC Model:

- *Potential Emotional States:* These represent the particular classes of emotions the current situation is likely to provoke according to the learner's inferred attitudes, goals and standard of behavior. Additionally, the model predicts how strong those potential emotions are likely to be.
  - For instance, learners might be expected to feel angry if they get confused by a hint suggested by the system and then fail to solve a particular problem, while they would feel gratitude for the system if the hint helps them solve the problem correctly.

- *Active Emotional States:* The ABC Model classifies the ongoing active emotions of learners based on sensory inputs, system interactions and predictive models.
  - It stores these emotions, along with their *intensity*, that is, the distance between the learner's estimated threshold and the actual current value measured for the respective emotion – the greater this distance, the greater the intensity of the experienced emotion. The active emotional states are one component in the appraisal process discussed later.

Behavioral Patterns

There are two key dimensions of an individual that are most responsible for the way they act: *behaviors* and *personality traits.* Personality traits are persistent characteristics that are demonstrated often under specific circumstances or environments. Because they define habitual patterns of thought and emotion, they provide a foundation for predicting behavior. Personality traits seldom change over time but these can trigger different behaviors under different circumstances and emotional states. The ABC Model incorporates representations of the personality traits and learner objectives, which combined with the affective states, predict the behavior of a learner under given circumstances.

Personality Traits: The ABC Model follows the Five-Factor Model of personality (Conati et al., 2002), which structures personality traits as five domains:

1. *Openness:* the degree of intellectual curiosity, creativity, and a preference for variety.

2. *Conscientiousness:* a tendency to show self-discipline, act dutifully, aim for achievement, and plan behavior.

3. *Extraversion:* sociability and the tendency to seek stimulation in the company of others.

4. *Agreeableness:* a tendency to agree, be cooperative, and go along with others, as well as how important it is for a person to please others.

5. *Neuroticism:* the tendency to experience unpleasant emotions easily, such as anger, anxiety, or depression, indicating the degree of emotional stability and impulse control.

All these dimensions of personality are closely related to the expressional, logical, and emotional personification to varying degrees. Depending on the application being implemented, a different combination of these personality traits might be considered significant for the context of usage. However, it is advisable to use all the dimensions in the model, as some changes in the learner's states might go unnoticed leading to inaccurate predictions under certain conditions. Since the model states that these five factors form the basis of the personality space, one should be able to represent any personality as a combination of these factors.

Goals: Goals are used in problem solving or task execution in learning environments. The learners using an e-learning system have goals and objectives to carry out an associated task or to perform and learn at a certain level. There can be different types of goals depending on the application that is using the ABC Model. A few examples follow:

- Performance related goals with respect to a specific task in an application.
- Goals related to achieving a certain level of understanding of concepts.
- Goals to avoid making errors and mistakes
- Goals to find the most efficient solution to problems
- Goals to perform well with minimum assistance from the system
- Goals to perform better than other students
- Goals to have fun in the learning process

Each of these goals has a different degree of relevance to a particular learner, which leads to events having varying effects on the affective states of the learner (Elliott et al., 1999). The relevance is represented using the following *intensity variables*:

1. Importance to the learner
2. Effort
3. Anxiety
4. Arousal

These variables are assigned a value by using the user information and the personality traits, for instance, the domain knowledge and preference decide the importance and effort associated with a goal and anxiety and arousal depend on specific personality traits and past record.

Cognitive States

Cognitive states represent the state of a person's cognitive processes or their state of mind. They represent the way a learner thinks, perceives, remembers, or solves problems under given circumstances. Hence, they have a large effect on the way a learner assimilates and retains information while using an e-learning system. Similar to the affective aspects of a learner, cognitive aspects are characterized by two types of data: the learner's *cognitive profile* and the learner's *cognitive states*.

Cognitive Profile: The cognitive profile represents the habitual pattern of cognitive behavior exhibited by the learner and is constituted by the following:

- *Cognitive State Thresholds:* These represent the assumed cognitive "resistance" to the different cognitive states.
    - For instance, cognitive thresholds can model how easily the learner can get confused. The rate at which the system provides information to the learners might be beyond their

ability to grasp it. Information about the learner states is monitored and recorded to assign threshold values to the parameters associated with the state and the signals sent by the sensors used for identifying the cognitive state.

- *Cognitive State Decays:* These represent the extent to which cognitive states represent self-sustaining processes or how long the learner persists in the states.
    - As with emotional class decays, it is very difficult to assign values to cognitive state decays. They can be inferred from past trends, for instance, some learners get bored while they are being taught a subject they are poor at and they pay less attention to a system that inhibits the learning process. A system typically responds to such a situation by attempting to refocus the learner's attention. The time span over which the system needs to take such measures can be observed and a decay rate can be assigned accordingly.

Cognitive States: Two types of cognitive states are stored in an ABC Model:

- *Potential cognitive states:* These represent the particular cognitive states the current situation is likely to provoke according to the learner model. For instance, learners might be expected to feel drowsy if they are currently getting bored and confused by a topic being taught by the system.

- *Active cognitive states:* The ABC Model classifies the ongoing active mental states of learners based on sensory inputs, system interactions, and predictive models. It stores these states, along with their *intensity*, that is, the distance between the learner's estimated threshold and the actual current value measured for the respective mental state – the greater this distance, the greater the intensity of the experienced state.

Based on the information representing the learner's affective, behavioral, and cognitive states, the ABC Model predicts how a learner is going to behave on the occurrence of an event. The next section describes the various types of user information required to build the user model and the ways of collecting the information.

User Information

The information about the user forms the basis of a user model and is a primary means of characterizing it. User models are created by processing different types of user information such as beliefs, characteristics, preferences, objectives, etc. The raw information about the user is processed to create the ABC Model. The user information can be classified into five categories: characteristics, capabilities, preferences, domain knowledge, and goals. Each of these categories is briefly described in this section followed by some means of gathering the information about the user.

Types of Information

Characteristics: These are normally captured within a profile of the user, such as gender, age, interests, and personality traits. This information helps in predicting the user's behavior under specific circumstances and aids the agent in making better decisions. This information can also be relevant to infer other more specific information such as preferences.

Capabilities: Some systems need to model the capabilities of their learners (e.g., the ability of the learner to understand a recommendation or explanation provided by the system). Modeling capabilities include modeling human learning, memory, and cognitive load limitations, which would allow a system to adjust the length and content of explanations as appropriate to ensure the learner is capable of assimilating it.

Preferences: These are used in interactive systems to make suggestions for the learner or in interface agents to select the information that is most relevant to the learner. The preferences often help the system recognize any tendencies of a learner toward particular options or solutions, which, in turn, helps in estimating the mistakes and errors made by the learner.

Domain Knowledge: These represent the learner's beliefs about a specific domain of knowledge. The knowledge of the concepts and terms the learner understands, allows the system to produce responses incorporating those concepts and terms while avoiding the concepts that the learner might not understand. This type of content is relevant for intelligent learning environments, which aim at considering the learner's state of knowledge to facilitate generation of explanations.

Goals: Goals have already been described briefly in the previous section on behavioral states. The effort that a learner puts in depends on the goals, and hence, the need to store information on these is an immediate result of the need to support the learner to adequately achieve their tasks and performance level. We describe later in this chapter how these goals and objectives affect the desirability of the events in an e-learning environment which in turn helps in modeling the learner's state.

Methods of Gathering Information

There is no absolute method for collecting the above-mentioned types of information; instead, a combination of methods should be used. This helps in gathering different segments of information, which can be combined together and then classified accordingly to generate a holistic model of the learner. The following are some of the methods that can be employed for this purpose:

- **Survey Forms:** These are often seen as a quick and easy means of collecting valuable learner information. These allow information collection in an objective and standardized manner, which helps in direct storage, usually without any need of further processing. For example, the learners can be asked about their specific traits on a particular scale and they can be asked to choose their preferences from a list. This is not usually possible with open-ended questions having a subjective nature since they can generate huge amounts of data. This makes their handling and processing a complicated task as the relevant information needs to be extracted from the learner's responses. This, survey forms should be used mostly for specific information about the learner, for instance, the characteristics, likes, dislikes, and basic objectives.

- **Learner's Academic Records:** These often reveal a huge amount of information about the academic background of the student learner. The knowledge of academic background helps in analyzing the familiarity the learner has over specific domains covered by the learning environment and, to some extent, describes the learner's relative ability to grasp different content. Learner records also help in estimating a range in which the learner can perform, which can be used to deduce the learner's objectives and expectations associated with the e-learning system.

- **Pretest and Questionnaires:** These are used to evaluate comprehension and deduce high level goals while using the e-learning system. The pretest can have different sections aimed at assessing the learner's skills related to various domains like mental ability, problem solving, and initial understanding of different subsections of a topic. Some questions can be specially designed in order to get information on the personality traits of the learner. Depending on the learner's performance, a basis can be created for the goals, focus areas, and most frequent errors.

- **Sensors:** The ABC Model gathers information about the learner's affective and cognitive states by the use of various sensors. These can be based on audio, video, infrared signals, thermal imaging, heart rate, respiration rate, etc. Passive sensors can be used to sense learner behavior unobtrusively, avoiding any negative impact on learning process (Sottilare et al., 2012). Sensors send signals to the learner model indicating the current state being experienced by the learner and this information is used to infer profiles related to emotional and cognitive states.

Events

In the context of the ABC Model, an event is an activity or a happening that can affect the realization of learner's goals and has the potential to modify the learner's affective and behavioral states, either directly or indirectly. Events can result from the actions of either the learner or the system, and hence, can be of two types based on the source of origin: *learner-generated events* and *system-generated events*. Based on their effect on the realization of the goals and objectives, events can be classified into the following two categories (Martinho et al., 2000):

- **Desirable Events:** events that lead to or facilitate the realization of goals and objectives.

- **Undesirable Events:** events that prevent or inhibit the achievement of goals and objectives.

In both cases, the degree of desirability is proportional to the importance of the goal and the degree to which the event contributes to or impedes the achievement of the goal.

Relation Between Events and Learner Model

The ABC Model recognizes the emotions based on the OCC cognitive psychological model of emotion, which considers 22 categories of emotions. Relevant categories of emotions are selected depending on the e-learning application for which the model is being used. Some of the emotional and mental states most relevant to e-learning environments are joy, distress, fear, disappointment, surprise, anger, boredom, confusion, attention distraction, and shame.

The events are assigned a desirability value based on their predicted effect on achieving goals, which is directly related to the learner model states. The desirability helps the system decide how a learner is expected to react to a particular event, under a given state. For example, an event inhibiting the achievement of a goal will be much more undesirable for a learner who has been ascribed a high value for the neuroticism trait and is already in a distressed emotional state compared to a happy learner with a positive attitude. According to OCC, joy and distress are elicited when a person focuses on the desirability of an event in relation to the individual's goals. Joy occurs when a person is pleased about a desirable event that takes place and distress when that person is displeased about an undesirable event. For instance, for learners who have the intention of pleasing the teacher and their parents, obtaining a good grade is a desirable event. Similarly, different cognitive states become active depending on the desirability of an event.

It is necessary to determine the learner's goals in order to verify the desirability of events. Students who have a learning goal are oriented toward developing new skills and abilities, and try to understand their work, improve their level of competence, and learn new things. When learners have performance goals, they want to demonstrate that they have the associated abilities. They feel successful when they please the teacher or do better than other learners, rather than when they understand something new.

Moreover, different classes of emotions are elicited depending on the source of an event. For example, an event that facilitates the learner's goals can generate positive emotions such as joy for the event, *gratitude* toward the system for system-generated events, or *pride* in the case of a learner-generated event. Corresponding negative emotions for an event inhibiting one's goals are *anger* and *shame*.

Appraisal Mechanism

A user modeling system needs to have a mechanism to appraise emotion-inducing stimuli, so that the affective and cognitive states are predicted with a high level of accuracy. People have a perception of the world and this leads to activation of emotions. The appraisal structure makes use of inductive and abductive reasoning over these perceptions. Induction allows inferring the conclusion from the premise with a high probability. For example, if learners have a tendency to behave in a particular way while experiencing some specific states, then the system predicts that behavior under similar circumstances in future. On the other hand, abduction allows inferring a premise as a plausible explanation of a consequence. For example, if undesirable events result in particular learner states, then these states arising after the occurrence of an uncategorized event can be used to classify it as undesirable. According to OCC theory, emotions are elicited from three different perspectives: consequences of events happening in the world, action of agents, and aspects of objects existing in the world. Based on the learner information, the ABC Model attempts to predict how the learner perceives different events. The predicted learner perception is used to infer the changes expected in the learner's affective, behavioral, and cognitive states on the occurrence of an event.

Desirability of events is estimated with respect to the learner's goals and this may elicit *consequence of events* emotions like satisfaction and distress. Appraisal of actions, whether of the learner or the system is done with respect to the learner's predicted standard of behavior and this may elicit *action of agents* emotions like admiration and remorse.

The ABC Model keeps a record of the past changes in the learner's affective and cognitive states, which is used to ascribe the potential states. Based on these and the information about the learner, the system makes predictions regarding the learner's expected behavior. This predicted behavior is compared with the actual observed behavior of the learner and depending on the level of conformation between the two, appropriate modifications are made to the learner model and the prediction models. Now based on the observed events or actions and the learner's current state, the system infers the changes activated by the event and the further actions expected to result from these changes. These inferences are then stored in the learner's model to be used for appraisal of future events. The system appraises events by assigning them a relevance related to the learner's goals and then estimates them using the OCC theory of emotions to evaluate each affective and cognitive parameter. This can be done using machine learning and pattern matching techniques. Predictions regarding the learner's behavior are also made using machine learning techniques based on the affective state of the learner and the information in the learner model.

Now based on the observed changes resulting from an event or an action and the knowledge of how the learner behaved under similar situations in the past, the system incorporates pedagogical strategies and provides suggestions to help the learner acquire states that enhance the learning process (D'Mello et al., 2009). For problem solving environments, the ABC Model keeps track of the possible steps at each stage and if the learner solves a problem incorrectly, then the system appraises the point of error in the solution and highlights it to the learner (Balakrishnan, 2011). When the learner makes an error, the ABC Model records the learner's states and associates them with the error. This helps in identifying the potential mistakes the learner might make in specific states, so when such a state arrives in the future, the system helps the learner to avoid making the mistake by producing hints.

Discussion and Recommendations for Learner Modeling

The ABC Learner Model framework described here integrates inductive and abductive reasoning to develop a learner model for predicting the learner's affective, behavioral, and cognitive states in an e-learning environment. We have described a convenient approach for the implementation of learner models based on the affective, behavioral, and cognitive states of the learner. It allows the system to predict learners' future ABC states based on their personal information and cognitive evaluation of events that elicited a specific emotion or behavior. The ABC model appraises events in terms of desirability and learns about the events that elicit specific emotions, and hence, decides the appropriate tactic to apply to enhance learning. Being independent of domain and not involving any predetermined assumptions, the proposed framework can be integrated into any e-learning environment. The affective and cognitive profiles can be used to focus on specific challenging states of a learner demanding special attention. Incorporating predicted future states allows the system to modulate learners' critical behavior in advance by giving appropriate suggestions.

GIFT, a service-oriented framework of tools, methods, and standards to aid computer-based tutoring systems described in previous chapters, can benefit by incorporating aspects of the ABC Model into its framework. The various types of information required to create the learner models can support GIFT's Domain Module in assessing the learner's performance and providing relevant content as feedback. GIFT's Learner Module functions to determine the learner's affective and cognitive states and this module can use the method in the ABC Model for predicting learner states. The ABC Model observes and analyzes learner performance over a period and creates affective and cognitive profiles that have threshold values and decay rates associated with the states in consideration. Incorporating these parameters can assist the learner module in determining learner states more effectively.

The ABC Model's appraisal mechanism predicts the changes resulting from an event and facilitates suitable interactions to enhance the learning process. GIFT's Pedagogical Module decides when feedback needs to be provided to the learner, and hence, can benefit from the ABC Model's appraisal mechanism. The appraisal mechanism can also support the trainee module for predicting the learner's state. Moreover, the Assessment Construct of GIFT, which is used for post-hoc analysis, can utilize the learner's affective, behavioral, and cognitive profiles created by the ABC model.

References

Adam, C., Herzig, A., Longin, D. (2009): *A logical formalization of the OCC theory of emotions.* Synthese, Volume: 168, Issue: 2, Publisher: Springer Netherlands, Pages: 201-248.

Balakrishnan, A. (2011): *On Modeling the Affective Effect on Learning.* MIWAI 2011, LNAI 7080, pp. 225–235. Springer, Heidelberg.

Conati, C., Zhou, X. (2002): *Modeling Students' Emotions from Cognitive Appraisal in Educational Games.* ITS 2002, LNCS 2363, pp. 944–954, 2002.© Springer-Verlag Berlin Heidelberg.

D'Mello, S., Craig, S., Fike, K., Graesser, A. (2009): *Responding to Learners' Cognitive- Affective States with Supportive and Shakeup Dialogues*. Human-Computer Interaction, Part III, HCII 2009, LNCS 5612, pp. 595–604, Springer, Heidelberg.

Elliott, C., Lester, J., Rickel, J. (1999): *Lifelike pedagogical agents and affective computing: an exploratory synthesis.* In Mike Wooldridge and Manuela Veloso, editors, AI Today. Springer - Lecture Notes in Artificial Intelligence.

Graesser, A. C., Jackson, G. T. & McDaniel, B. (2007). *AutoTutor holds conversations with learners that are responsive to their cognitive and emotional states*. Educational Technology, 47, 19-22.

Jaques, P.A., Vicari, R., Pesty, S., Martin, J. (2011): *Evaluating a Cognitive-Based Affective Student Model* . Affective Computing and Intelligent Interaction - 4th International Conference, ACII.

Kass, R., Finin, T. (1998): *Modeling the User in Natural Language Systems.* Computational Linguistics, Volume 14, Number 3.

Kobsa, A., Pohl, W. (1995): *The User Modeling Shell System BGP-MS.* User Modeling and User-Adapted Interaction, 4(2):59–106.

Martinho, C., Machado, I., Paiva, A. (2000): *A Cognitive Approach to Affective User Modeling*. In: Paiva, A. (ed.) IWAI 1999. LNCS, vol. 1814, pp. 64–75. Springer, Heidelberg.

Ortony, A., Clore, G., Collins, A. (1988): *The Cognitive Structure of Emotions*. Cambridge University Press, Cambridge.

Sottilare, R.A. and Proctor, M. (2012). *Passively Classifying Student Mood and Performance within Intelligent Tutors.* Educational Technology & Society, 15 (2), 101–114.

CHAPTER 8 –The Need for Empirical Evaluation of Learner Model Elements

Heather K. Holden and Anne M. Sinatra
U.S. Army Research Laboratory (ARL) - Human Research and Engineering Directorate (HRED)

Introduction

The data stored in a learner model are used by an ITS to adapt and customize instruction based on the learner's state of cognitive and affective knowledge. Ideally, this model would include information about the learner's individual characteristics, past and current competency, performance, cognition, affect, behaviors, etc. The higher the level of functionality and capability of a learner model to interpret and accurately classify the comprehensive knowledge of an individual learner, the better the ITS can adapt to the individualized needs of the learner. For expert human tutors, this process is easy since they have the natural ability to interpret and assess the learner's current and predicted state of readiness for instruction. However, equipping learner models with such capabilities is a computationally complex problem ITS researchers have been trying solve over the last 15 years. The primary sub-research areas of learner modeling research include, but are not limited to, learner state, cognitive modeling, affective modeling, individual differences, behavioral and physiological sensing, and performance assessment.

Most learner modeling research is conducted within academic populations (primarily K–12) and well-defined, domain-specific ITSs, such as mathematics and physics. The next-generation ITSs aim to be more inclusive of adult learners and job-related training; however, little is known about the transferability and validation of previous research findings as well as the investigation of other useful learner aspects that scale beyond academia. While great strides have been made among learner modeling research, there are several factors that limit the future progression/development of comprehensive learner models: (1) the lack of understanding of the impact and interaction effects of learner model elements; (2) the lack of reusability and transferability of learner models between ITSs, domains, and populations; (3) the nonexistence of the measures from user models/modeling of which learner models/modeling is a subset; and (4) the lack of standardization for learner model development and structure.

These limiting factors can be addressed through the process of empirical evaluation in future learner modeling research and learner model development. For years, these issues have been ignored by the user modeling community at large (Adikari & McDonald, 2006; Glavinic & Granic, 2008; Granic & Adams, 2011; Johnson, 1994; Kobsa, 1994) due to the constraints surrounding the needed experimentation. However, GIFT, an experimental testbed that can accommodate such evaluations and comparative analyses, is now available to conduct such experimentation realistically and affordably.

The purpose of this chapter is to provide justification for the need of empirical evaluation by taking lessons learned from the human-computer interaction (HCI) perspective of user modeling research. Moreover, we present guidelines and suggestions on how to conduct such evaluations using GIFT. Specifically, this chapter is divided into three sections: (1) the current understanding of learner model elements; (2) the incorporation of the missing link of HCI user modeling research; and (3) suggestions on how to conduct experimentation using GIFT toward the development of standardizing and generating comprehensive learner models capable of accommodating user and domain diversity.

Current Understanding of Learner Model Elements

The content within learner models is generally categorized in two parts: domain-specific or domain-independent information (i.e., learner-specific characteristics [individual differences]) (Abdullah, 2003; Gonzalez, Burguillo & Llamas, 2006). Domain-specific information reflects the learner's state and level of knowledge or ability within a particular domain. This type of information primarily includes historical competency (domain knowledge and skills measured over time), misconceptions, problem-solving strategies, etc. Most learner models, particularly those of first-generation ITSs, are concerned with modeling this type of information because this allows the model to be more generalized across multiple populations. While this information is useful, it alone is not sufficient for providing the highly adaptive individualized training. Domain-independent information consists of all relevant characteristics of an individual learner and can include, but is not limited to, the following elements: learning goals; cognitive aptitudes; measures of motivational state; learning preferences (including styles and personality); interest; demographics; past performance and competency (non-domain-specific); behavioral/psychological measures; cognitive and affective dimensions; and personal control beliefs (including general self-efficacy; locus of control). These individual difference variables are significantly different between learners and, collectively, are not the same for any two learners.

To accurately classify a learner's state and performance at any given time, the learner models must have all-inclusive understanding of learner's cognition and affective states, influential individual difference characteristics, and performance. One area of learner modeling research is dedicated to understanding the influence of and interrelationship between domain-independent information (e.g., learner-specific characteristics) and how it can be best used in conjunction with the domain-specific information to optimally classify a learner state and performance. Understanding impact and interaction effects of individual learner model elements takes "big data," recurring empirical evaluation and experimentation, and the ability to dynamically incorporate multiple models and modeling techniques simultaneously. However, learner models have limited reusability since they are typically developed standalone and tightly coupled within the specific ITS within which it is integrated. Most of these systems typically can only accommodate one well-defined academic domain (i.e., mathematics, physics, computer science), resulting in the lack of standardization of learner model elements and ideal learner modeling techniques. The review of HCI user modeling research shows the same limitations are present in that area of research. While the approach to user modeling is different, lessons can be learned from the HCI user modeling research area. The next section discusses how ITS learner modeling research can be enhanced by leveraging some of the research of its parent research area, user modeling (from the HCI prospective).

The Missing LINK: The Human-Computer Interaction (HCI) User Modeling Perspective

There are several social and economic factors influencing the evolution of technology; however, technology progression is directly correlated to changes in user requirements. Consider the evolution chain of the personal computer (i.e., desktop to laptops to netbooks to tablets and smartphones) as an example. This progression was accomplished by user's desire to have these devices more portable, faster, and useful to accommodate users with diverse computing purposes. User requirements for the successful and beneficial usage of ITSs are also changing as their need to account for learner and domain diversity increases. Long gone are the days in which user-initiated and user-selected adaptation techniques, such as completing preference menus and editing profile files, are sufficient for personalizing interactive computer systems (Kobsa, 1994). In most cases, especially ITSs, learners (users) do not have the necessary knowledge about the subject area, their own errors, or the system's adaptive abilities to select adaptation preferences. Moreover, adaptation methods in current and future ITSs dynamically occur and are too numerous for learners to customize each potential adaptation path.

User and learner modeling research have the opportunity to significantly enhance the adaptive capabilities of intelligent, interactive interfaces and learning environments; however, there is a rather large disconnect between these two research areas limiting their forward progressions. Learner models and modeling is a subset of user models and modeling (Self, 1988). User models, like learner models, contain the system's assumptions about all user characteristics that are relevant for tailoring system behavior to accommodate the individual user. Furthermore, both user and learner modeling share common tasks including (1) initializing the user or learner model; (2) drawing assumptions about the user or learner based on system interactions and updating the user or learner model accordingly; and (3) supplying other system components with assumptions about the user or learner, as needed. While user and learner modeling share common performing functions, there seems to be a rather large disconnect between these areas. They differ in the following ways:

- **Primary Area of Research**
    - User modeling is a subdivision of HCI
    - System goal is to build useful and usable systems
    - Learner modeling is viewed as a subdivision of AI. This terminology is essentially used for the primary user (i.e., student, learner, trainee, pupil, etc.) of an ITS and other learning environments.
    - System goal is to build systems that portray intelligent behavior
- **Model Content**
    - Both user and learner models can contain personal data associated with a specific user/learner including demographics, past experience, goals, interests and motivation, knowledge and skills, preferences, etc.
    - User models also include and emphasize users' system preferences and dislikes, behaviors and interactions with the system, system acceptance (including perceptions, satisfaction, and usability), and general technology acceptance.
    - Learner models also include and emphasize learner's cognitive and affective states, domain competency and self-efficacy, cognitive aptitudes, etc.
    - There is no current standardization on how to structure and employ these models; therefore, not all current models contain the above information.
- **Adaptation Techniques**
    - User modeling focuses on modifying/adapting the system's interface design based on the user model. Considers other elements of HCI such as usability and user-centered design.
    - Learner modeling focuses on modifying/adapting instruction based on the learner model (not including the change to the physical user interface, but may change the interface feedback via agent, text, audio, etc.).

- **Presence in the Literature:**
  - It is rare to find representation of ITSs and learner models in user modeling research, and it's even more scarce to find reference to user modeling in ITS and learner modeling research. Basically, these two research areas and system development are conducted entirely independently (Johnson, 1994).

    Johnson (1994) suggested that regardless the past neglect of combining the research and development of HCI- and AI-based user models, the two communities will come together in the near future due to the increase of intelligent interface and agents; however, 20 years later, gap and disparities still remain. If both communities understand the importance of optimizing the user's system interactions and have had significant progress within their respected fields, why is the gap between these two areas still as widespread as it was 20 years ago? Why does this issue still matter?

Returning to the notion of technology evolution being driven by user requirements, the need for understanding how an interactive system can dynamically capture/model users' needs and adapt its interaction accordingly has become more vital due to the increase in range and complexity of user requirements for such systems (Granic & Nakic, 2007). In order for ITSs to optimize learning experiences and system intelligent behavior, a greater understanding of the interaction between the learner and system is needed. We can no longer ignore the needed synergy between the learner's learning process (inclusive of individual differences) and the learner's interaction with the learning application (i.e., ITS) (Granic & Adams, 2011; Squires & Preece, 1996). HCI user modeling research can provide the ITS learner modeling community with a potential solution to ascertaining such an understanding, which will directly attribute to better understanding the impact and interactions of currently researched learner model elements.

User Modeling in Human Computer Interaction (HCI) Explained

The general goal of HCI is to facilitate the development of systems that are enjoyable and easy to use. User modeling in HCI research was originally aimed at investigating the different types of user models and their role in supplying information for designers throughout various stages of the system development process (Johnson, 1994). Over the last 10 years, HCI research has realized that understanding users' needs is at the core of successful interactive technology design and adoption. Therefore, this research area has expanded to investigating user-centered, user-sensitive, and learner-centered design approaches toward the development of transparent interfaces and flexible interactions that can account for user diversity (Glavinic & Granic, 2008). Thus, such research has extended its objective to gaining a thorough understanding of the cognitive, perceptual, and motor components of user interactions with interactive systems (Olson & Olson, 2003).

User-sensitive design places equal focus on user requirements and the diversity of such requirements among all intended users (both typical and extraordinary) (Granic & Adams, 2011). Learner-sensitive design (Soloway et al., 1996) expands user-sensitive design by accounting for learner's unique needs, objectives, knowledge, abilities, and other learner-specific characteristics. Since it is known that one single interface design will not satisfy every user, HCI user modeling research looks to intelligent user interfaces (IUIs) as a means of (1) providing more individualized and personalized interactions, (2) enabling adaptation of interface behavior to match user individual characteristics (adaptive systems), and (3) enhancing system acceptance, usability, flexibility, and attractiveness (Granic, 2008b; Hook, 2000).

Adaptive, IUIs rely on the use of user models, which contain a collection of information and assumptions about particular users that guide the adaptation process of the system for an individual (Kobsa, 1995), thus an intelligent system's behavior strongly depends on the impact of user individual characteristics on interaction with the system (Granic & Nakic, 2007; Magoulas, Chen & Papanikolaou, 2003). This is similar to the use of learner models in ITSs. Like ITS research, HCI research has acquired inconsistent results on the impact of individual differences on user performance; however, the underlying cause in HCI research attributing to user performance with a particular system largely depends on the system alone (Granic & Nakic, 2007). In HCI user modeling research, elements of users' acceptance, preferences, usage behavior, perceptions, perceived usability and usefulness, and attitudes toward the system and computers in general are considered as a part of modeling users' system interactions.

ITS learner modeling research can leverage some of these elements to ascertain a clearer distinction between factors influencing a learner's cognitive and affective knowledge during the learning process and factors that are directly linked to system interaction and usage behavior. With this concept in mind, several HCI researchers have investigated the link between users' individual differences and their usage of e-learning applications (as ITSs fall under the same umbrella of educational technology). Adams (2007) evaluated eight hypothetical criteria e-learning systems need to accommodate individualized student learning against five e-learning platforms. Accessibility and student modeling for user diversity were the weakest points among all cases. The Cognitive Tutor Authoring Tool (CTAT) was found to address the most criteria among the five platforms; however, the difficulty and time involved in developing a cognitive model limits its universal usability (Adams, 2007). An experiment conducted to investigate the existence and level of interaction among users' individual differences and learning outcomes through the use of an e-learning application also garnered interesting results. The individual differences that were evaluated within the experiment include both *personal* user characteristics (i.e., intelligence, emotional stability, extraversion, mental stability, experience) and *system-dependent* user characteristics (experience using computers and Internet, motivation to learn programming, expectations from e-learning, and background knowledge material to be learned). The study found significant correlations between mental stability and motivation, and emotional stability and expectations from the system; however, only learner's motivation to learn programming and their expectations of e-learning had a significant impact on the knowledge acquired through their system interaction (Granic & Nakic, 2007). A follow-on study also found motivation to learn and expectations about e-learning (both for e-learning in general and the specific application) to significantly influence learning achievement (Granic & Adams, 2011). These findings support the need for researching the impact and interaction of individual characteristics (inclusive of system-dependent/specific characteristics) and how users' expectations of the system can impact their successful system interactions.

For learners, their interest and motivation to learn pertains to their willingness, direction, intensity, and persistence of learning-directed behavior. It influences their choices during learning activities as well as cognitive engagement during instruction and training (Schultz, Alderton & Bordwell-Hyneman, 2011). The level of a learner's intrinsic motivation, goal orientation, and need for achievement are also directly related to an overall motivation to learn and has been shown to be directly related to learning performance and other learning outcomes (Schultz et al., 2011). Furthermore, learners' self-efficacy beliefs are also related to their motivation to learn, learning, performance, and job performance (Glavinic & Granic, 2008) and have been shown to influence learners' decision making during instruction and training (Soloway et al., 1996). These aspects should be contained within the learner model structure; however, research assessing the influence of learners' motivational characteristics on outcomes and their relationships to other individual difference variables is practically non-existent. Although ITS research has found interrelationships among learner-specific characteristics (for examples, personality and cognitive abilities [(Kobsa, 1994; Schultz et al., 2011)], and learning styles and cognitive traits (Graf, Liu, Kinshuk, Chen & Yang, 2009), current learner models have a limited capability to account for individual differences as explanations of learner's cognitive and affective knowledge. Based on the studies previously mentioned, user models also have limited capabilities for accounting for individual characteristics.

Another important area of HCI user modeling research to consider is end-user technology acceptability and adoption. While technology has been deemed as the “salvation” to education by providing individualized learning, it rarely meets this “broad expectation” (Healey, 1999). Authoring tools and shells, such as CTAT, are designed to accommodate teachers by supporting them in the development of a series of ITSs; however, the ITS adoption and usage for tutoring in real classrooms has been a slow progression. HCI researchers attribute this slow adoption rate to the fact that ITS interaction mechanisms have not been accompanied by an adequate user interface design (Granic, 2008a).

In addition to its focus on user expectations and requirements, technology acceptance also considers user perceptions of a specific technology’s usefulness and usability. Liu, Laio, and Peng (2005) found significant evidence that learners of e-learning applications have two identities, one as a system user and the other as a learner, and both identities are influenced by the “flow” (concentration) and perceived usefulness of the e-learning system (Liu, Liao & Peng, 2005). A recent meta-analysis of research found solid evidence that supports perceived usefulness is the strongest predictor of a learner’s adoption of an e-learning technology (Sumak, Hericko & Pusnik, 2011). Usability evaluation is an important role in user interface design; however, the number of usability studies on e-learning is limited and the consolidated evaluation methodology for e-learning is non-existent (Ardio et al., 2006; Costabile, Marisco, Lanzilotti, Plantamura & Roselli, 2005; Granic, 2008a). These studies mention the need for further research and empirical evaluation of usability assessment of e-learning applications. Usability assessment and measurement are always among HCI’s approach to investigating the interactions between users and the system; determining its value in modeling interactions between learners and ITSs is a valuable factor to consider. Squires and Preece (1996) affirm that “there is a need to help evaluators consider the way in which usability and learning interact” (Squires & Preece, 1996); Costabile et al. (2005) also argue that the usability of an e-learning application can directly affect learning (Costabile et al., 2005).

Usability evaluation can be measured by objective performance metrics of efficiency and effectiveness as well as the user’s subjective assessment of the system usage. These objectives quantify user performance, satisfaction, and terms by which they find the system acceptable. Adikari and McDonald (2006) constructed a science-oriented research design to test the value of incorporating conceptual user modeling and usability modeling into product requirement specifications for improving design (Adikari & McDonald, 2006). For authoring shells and ITSs, such evaluations can help identify the exact problems of a particular system (Granic, 2008a) and help separate problems/issues pertaining to the learning process.

The ITS and learner modeling community can benefit from these aspects of HCI user modeling research. By combining the same evaluated user modeling elements of perceptions toward learning, learner models could potentially increase explanation of states, performance, and system behavior. Little ITS research has been done in this area; however, preliminary findings have shown that there is a significant relationship between learners’ acceptances of pedagogical agents, or virtual tutors, embedded within a learning environment and the learners’ acceptances of the learning environment itself (Adams, 2007). A prior study also identified links between students’ behaviors with a tutor and their attitudes and perceptions (Healey, 1999). Research blending these areas will also be beneficial to HCI user modeling as both HCI and ITS user modeling research areas have the same issues of no standardization, the inability to accommodate individual differences, and the need for empirical evaluations to validate modeling elements. HCI user modeling has expresses the need for empirical research over the last 20 years, but this is a concept that has become more apparent recently within the ITS community.

Recommendations and Future Research: Towards the Development of Reusable and Standardized Learner Models

While much research is needed to investigate the transferability of previous findings, future ITS researchers and developers should consider the following: model development and evaluation of a few elements at a time to identify interrelationships between elements and their influences on learner state; controlled experimentation (including sensor validation and comparisons to self-reported data and user-experience post-experiment interviews); and increased collaboration and data sharing.

Practical Implications for Researchers

The field of ITS learner modeling research has close ties with other adaptive computing in fields such as user modeling, HCI, and AI. A common problem among these fields is the limited amount of empirical evaluation associated with the adaptive systems that they produce (Chin, 2001; Mulwa, Lawless, Sharp & Wade, 2011; Weibelzahl & Weber, 2002). One of the main reasons for this is the inherent difficulty of separating out the pieces of an adaptive system. Many of the existing ITSs are tied to one specific domain (e.g., physics, mathematics) and the learner model used within it is closely tied to the system. The cost, time, and difficulty that goes into developing these systems results in inflexibility and an impracticality of many experimental evaluations. In many cases, a non-adaptive, or control, group may make very little sense or be difficult when trying to evaluate the impact of differing adaptations in an empirical manner (Mulwa et al., 2011).

An additional challenge to ITS research is that it is difficult to separate out the user's learning outcomes from their ability to understand and use the system. One approach that has been taken to studying adaptive systems and user models has been to layer evaluations. Rather than trying to examine the entire system, individual pieces are evaluated and empirical studies are run at each part to make sure that it is effective (Mulwa et al., 2011). This approach is a step in the right direction for making it more practical to effectively evaluate the impact of adaptations and information stored in user or learner models.

It is of great importance to researchers, specifically in the field of ITSs, to ensure that information that is contained in the learner model and to which it is being adapted to actually provides a benefit. One way to do this is by examining the specific content of what is included in individual learner models and its impact on learning outcomes. Presently, each individual ITS uses its own combination of domain-independent components in the learner model (e.g., motivation, personality scores, cognitive measures). Many learner models may not even include information pertaining to computer familiarity and system acceptance, which have previously been found to be heavily correlated to overall performance in HCI and user modeling research (Granic, 2008b). Therefore, one of the next necessary and useful steps in ITS research is to empirically evaluate the impact and interactions of specific learner model elements. Through this empirical research, a set of useful and standardized domain-independent learner model components can be developed.

As it stands now, when researchers do examine the individual difference elements within their learner models, it is within specific domains with limited generalizability (Granic, 2008b). Studies such as those reported by Granic (2008b) have begun to examine which learner model elements have an impact on performance outcomes in a specific ITS system. The next step is to continue generating empirical studies that examine the learner model elements and their utility, and then build a body of knowledge, which can be examined as a whole looking for commonalities in the useful elements between domains. This examination can then lead to a generalizable learner model, which will contain useful information that can be applied in ITSs of varying domains.

While many of the components included in learner models have been shown to impact performance in traditional and classroom environments, they may behave differently when computer delivery is added to the equation. Therefore, it is important to conduct studies on these individual difference components within a computerized tutoring environment to see which one impact learning outcomes.

Experimental Design Recommendations

In general, there is a need for more empirical evaluations of learner model elements. There are a number of different steps that can be taken to increase our knowledge about the impact and interactions of learner modeling elements:

Literature review and meta-analysis. A thorough and formal literature review of current empirical research into learner model elements is necessary. It can provide an overview of the different techniques that are used to assess ITSs and learner model elements. It may also lend insight into which elements are commonly included in learner models and which ones have been found to be effective. A meta-analysis could give researchers a better picture of the types of domains that have been examined (e.g., algebra, physics), the number of tutors that have been assessed in each area, and what elements were included in those user models. The meta-analysis would show which elements of learner models were consistently helpful between these domains and which ones are domain specific. This would then give researchers a direction to take when generating specific experiments to test what elements matter in what situations. It also would give researchers a better understanding of potential interactions that exist between learner model elements. The meta-analysis and literature review will also highlight specific gaps in the literature and areas that have not received much attention.

More empirical evaluations. Granic (2008b) examined the different components of a learner model used in a computer programming tutor. Correlations were found between certain elements of the model (e.g., motivation to learn programming) and performance. However, personality factors did not have many correlations with performance (Granic, 2008b). More research of this type should be done with different domains, and different ITSs. Once a large body of empirical evaluation literature has been built up it, can be further examined to see what elements have utility throughout varying domains and which ones appear to be less generalizable. By expanding research in this manner, it will move our learner models to be more consistent with each other, and more easily comparable. GIFT is an ideal experimental testbed to use for such experiments.

Advantages of Using GIFT as an Experimental Testbed and Design Recommendations

A majority of learner modeling research has not focused on examining the same learner model in multiple domains. Since GIFT is a domain-independent framework, it allows teachers and researchers to design their content to work with it, rather than having to develop their own delivery system. One of the main benefits of designing content in this manner is that it will significantly reduce the amount of time and effort that would go into developing an ITS. This also allows for consistency between generated ITSs.

One of the intentions behind GIFT is to be able to easily interchange the pieces of a system, and even, components of the learner model. Further, an additional capability would be to provide a consistent structure for the development of ITSs. Therefore, GIFT is an ideal system to use for the development of ITSs and the empirical evaluation of learner model elements. A researcher can design a tutor with GIFT, and then use the architecture to plug in and hold constant the elements of the learner model to be tested (for instance, testing one condition where motivation level is adjusted, another where personality type is adjusted, and finally, one where both motivation level and personality type are adjusted, and measuring performance). This allows for the examination of the impact of individual learner model elements, and the possible interactions between them. In a custom system that is tied tightly to its content and learner model, these types of experiments would either be extremely difficult or impossible to complete. As GIFT's features continue to develop in the future, it will provide even more flexibility and granularity in the types of manipulations that researchers can conduct in their experimental evaluations of the components of learner models.

Conclusions

The learner model is a vital part of an ITS. The learner model often contains domain-independent information (such as individual differences) about the specific learner, and then adjusts instruction based on these differences. However, there has been very little research on the individual difference components that have been included in learner models, and there is no standardization of the models between systems. It is important for research in the ITS field to (1) examine the impact and interaction effects of learner model elements; (2) increase the reusability and transferability of learner models into different domains; (3) look to fields such as HCI for guidance into elements that may be useful within the learner model; and (4) begin to move toward standardization of learner models. GIFT provides an ideal testbed to use for affordable and efficient experiments into the impact and interaction effects of different learner model elements. Through further empirical evaluations and the use of GIFT as a research tool, the ITS field can move toward generating more comprehensive and consistent learner models that are highly generalizable between domains

References

Abdullah, S. (2003). Student Modelling by Adaptive Testing - A Knowledge-based Approach. Unpublished Disseration. The University of Kent at Canterbury (Dissertation).

Adams, R. (2007). *User Modeling for Intelligent Interfaces in E-Learning*. Paper presented at the Unversal Access in HCI, HCII.

Adikari, S. & McDonald, C. (2006). *User and Usability Modeling for HCI/HMI: A Research Design.* Paper presented at the International Conference on Information and Automation, Shandong.

Ardio, C., Constabile, M., De Marsico, M., Lanzilotti, R., Leviald, S., Roselli, T., et al. (2006). An Approach to Usability Evaluations of E-Learning Applications. *Universal Access in the Information Society,* 4, 270-283.

Chin, D. N. (2001). Empirical evaluation of user models and user-adapted systems. *User modeling and user-adapted interaction*, 11(1), 181-194.

Costabile, M., Marisco, M., Lanzilotti, R., Plantamura, V. & Roselli, T. (2005). *On the Usability Evaluation of E-Learning Applications.* Paper presented at the 38th Annual Hawaii International Conference on System Sciences, Hawaii.

Glavinic, V. & Granic, A. (2008). HCI Research for E-Learning: Adaptability and Adaptivity to Support Better User Interaction. In A. Holzinger (Ed.), *HCI and Usability for Education and Work* (pp. 359-376). Berlin Heidelberg: Springer-Verlag.

Gonzalez, C., Burguillo, J. & Llamas, M. (2006). *A Qualitative Comparison of Techniques for Student Modeling in Intelligent Tutoring Systems*. Paper presented at the 36th ASEE/IEEE Frontiers in Education Conference.

Graf, S., Liu, T., Kinshuk, Chen, N. & Yang, C. (2009). Learning Styles and Cogntive Traits - Their Relationship and Its Benefits in Web-based Educational Systems. *Computers in Human Behavior,* 25, 1280-1289.

Granic, A. (2008a). Experience with Usability Evaluation of E-Learning Systems. *Universal Access in the Information Society*, 7, 209-221.

Granic, A. (2008b). Intelligent Intefaces for Technology-Enhanced Learning. In S. Pinder (Ed.), *Advances in Human-Computer Interaction* (pp. 143-160).

Granic, A. & Adams, R. (2011). User Sensitive Research in E-Learning: Exploring the Role of Individual User Characteristics. *Universal Access in the Information Society*, 10, 307-318.

Granic, A. & Nakic, J. (2007). *Meeting User Individual Characteristics Through Adaptive Interface of an E-Learning System: An Empiracle Study Design.* Paper presented at the 29th International Conference on Information Technology Interfaces, Cavtat, Croatia.

Healey, D. (1999). Theory and Research: Autonomy in Language Learning. In J. Egbert & E. Hanson-Smith (Eds.), *CALL Environments: Research, Practice, and Critical Issues* (pp. 391-402). Alexandria, VA.

Hook, K. (2000). Steps to Take Before Intelligent User Interfaces Become Real. *Journal of Interacting with Computers,* 12, 400-426.

Johnson, H. (1994). Relationship Between User Models in HCI and AI. *IEEE Computers and Digital Techniques Proceedings,* 141(2), 99-103.

Kobsa, A. (1994). *User Modeling and User-Adapted Interaction*. Paper presented at the Computer-Human Interaction (CHI).

Kobsa, A. (1995). *Supporting User Interfaces for All through User Modeling*. Paper presented at the 6th International Conference on Human-Computer Interaction (HCI International).

Liu, S., Liao, H. & Peng, C. (2005). Applying the Technology Acceptance Model and Flow Theory to Online E-learning Users' Acceptance Behavior. *Issues in Information Systems,* 6(2), 175-181.

Magoulas, G. D., Chen, S. Y. & Papanikolaou, K. A. (2003). *Integrating Layered and Heuristic Evaluation for Adaptive Learning Environments*. Paper presented at the 2nd Workshop on Empirical Evaluation of Adaptive Systems held at the 9th International Conference on User Modeling.

Mulwa, C., Lawless, S., Sharp, M. & Wade, V. (2011). The evaluation of adaptive and personalised information retrieval systems: a review. *International Journal of Knowledge and Web Intelligence*, 2(2), 138-156.

Olson, G. M. & Olson, J. S. (2003). Human-Computer Interaction: Psychological Aspects of Human Use of Computing. *Annual Review of Psychology,* 54, 491-516.

Schultz, R., Alderton, D. & Bordwell-Hyneman, A. (2011). *Individual Differences and Learning Performance in Computer-based Training* (Technical note 1 Apr 2010-28 Feb 2011): NAVY PERSONNEL RESEARCH STUDIES AND TECHNOLOGY MILLINGTON TN

Self, J. (1988). Student Models: What use are they? In P. Ercoli & R. Lewis (Eds.), *Artificial Intelligence Tools in Education* (pp. 73-86). North Holland: Elsevier Science Publishers.

Soloway, E., Jackson, S., Klein, J., Quintan, C., Reed, J., Spitulnik, J., et al. (1996). *Learning Theory in Practice: Case Studies of Learner-Centered Design.* Paper presented at the SIGCHI Conference on Human Factors In Computing Systems: Common Ground, Vancouver, British Columbia, Canada.

Squires, D. & Preece, J. (1996). Usability and Learning: Evaluating the Potential of Educational Software. *Computing and Education,* 27, 15-22.

Sumak, B., Hericko, M. & Pusnik, M. (2011). *Factors Affecting the Adoption of E-Learning: A Meta-analysis of Existing Knowledge*. Paper presented at the The 3rd International Conference on Mobile, Hybrid, and On-line Learning.

Weibelzahl, S. & Weber, G. (2002). Advantages, opportunities and limits of empirical evaluations: Evaluating adaptive systems. *KI*, 16(3), 17-20.

CHAPTER 9 –On the Use of Learner Micromodels as Partial Solutions to Complex Problems in a Multiagent, Conversation-based Intelligent Tutoring System

Xiangen Hu, Donald M. Morrison, Zhiqiang Cai
Institute for Intelligent Systems (IIS)
University of Memphis (UM)

Introduction

More than 40 years since their origins in the early days of computer-assisted instruction in the 1970s (e.g., Carbonell, 1970), considerable progress has been made in developing "intelligent" (computer-based) tutoring systems that successfully scaffold learning in specific domains and for specific purposes (for a recent review, see Graesser, Conley & Olney [2012]). Frequently cited examples include Cognitive Tutor, which supports learning in algebra, geometry, and programming languages (Ritter, Anderson, Koedinger & Corbett, 2007); AutoTutor (Graesser, Olney, Haynes & Chipman, 2005; Graesser, Jeon & Dufty, 2008; Graesser, Lu et al., 2004), which does the same thing for college-level computer literacy, physics, and critical thinking skills; and the so-called "constraint-based" systems developed by Mitrovic's group in New Zealand (Mitrovic, Martin & Suraweera, 2007), which, among other topics, help students learn to program in SQL.

In spite of these advances at local research and development sites, the field has yet to produce a truly general-purpose system that is capable of supporting rapid development of high-quality applications across a broad range of domains. GIFT, currently under development at the U.S. Army's Learning in Intelligent Tutoring Environments (LITE) Laboratory, is intended to fill this gap. Citing Picard (2006), the developers claim that the availability and use of ITSs has been limited by the high cost of development, lack of reusability, lack of standards, and "inadequate adaptability to the needs of learners" (Sottilare, Brawner, Goldberg & Holden, 2012:1). These systems, they write, tend to be built as "domain-specific, unique, one-of-a-kind, largely domain-dependent solutions focused on a single pedagogical strategy." GIFT is presented as a solution to this problem. The modular framework and standards built into the system could "enhance reuse, support authoring and optimization of CBTS[3] strategies for learning, and lower the cost and skill set needed for users to adopt CBTS solutions for military training and education" (Sottilare et al., 2012:1).

The development of a general-purpose, domain-specific ITS framework is indeed an important goal, but numerous barriers block the way. These include (but are not restricted to) a lack of agreement in the ITS community about how the different components of an ITS ought to fit together; what the structure and content of the components ought to be; and how knowledge of the world is to be represented, both for experts and learners.

The argument made in this chapter assumes that a general-purpose system will at some point employ an open, multiagent architecture, meaning that core functions are carried out by more or less autonomous software agents united by a common ACL. Some of these agents will perform simple tasks (such as analyzing a learner's facial expressions), while others will take on more complex ones, such as generating appropriate responses to user questions. As an example, here we describe an autonomous software agent that produces a turn-by-turn analysis of a user's discourse moves on two dimensions: *relevance* and *novelty* (R-N). In the process, it builds what we call a *micromodel* of the learner's current state, including

[3] Here we use the term ITS to mean the same thing as a computer-based tutoring system (CBTS), a class of adaptive educational system (AES).

a relevance-novelty measure for single turns and for a series of turns.[4] This micromodel allows the R-N agent to make assertions about the learner's current and recent contributions to a conversation—assertions which may be broadcast generally or addressed directly to other agents, such as a conversation agent, a pedagogical agent, an agent responsible for constructing aggregate learner models from multiple micromodels, an agent that analyzes the effectiveness of instructional modules, or, where the ITS employs an "open" learner model (Bull, 2004; Bull & Pain, 1995; Kay, 2001; Mitrovic & Martin, 2002), an agent responsible for providing access to the learner model through the user interface.

The Standard Four-Component ITS Model

As noted elsewhere in this book, it is customary to identify an ITS as consisting of four major components, referred to as "models" – the learner model (sometimes called the "student model"); the expert domain model; the tutor model, and the user interface (Elson-Cook, 1993; Graesser et al., 2012; Nkambou, Mizoguchi & Bourdeau, 2010; Psotka, Massey & Mutter, 1988; Sleeman & Brown 1982; VanLehn, 2006; Woolf, 2008). In an ITS where the tutor is capable of mixed-initiative dialog with the user (Carbonell, 1970; Allen, Guinn & Horvtz, 1999; Graesser et. al, 2005), the tutor takes the form of an intelligent conversation agent, backed by a Dialog Advancer Network (Person, Bautista, Kreuz, Graesser & Tutoring Research Group, 2000). Figure 9-1 illustrates the relationship among these four components.

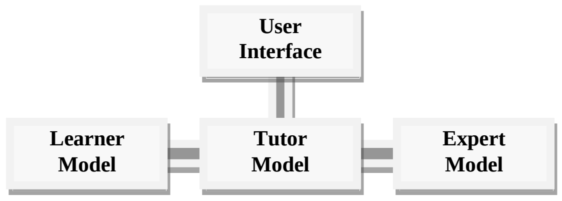


Figure 9-1. Standard ITS components

In general terms, the learner model represents what the tutor has established to be the learner's current level of knowledge, skill, and affective state, while the expert domain model represents the knowledge and skills the learner is supposed to acquire – and has therefore been called the "ideal student model" (Corbett, Koedinger & Anderson, 1997). Through interactions with the learner via the user interface, the conversation agent, playing the role of the tutor, seeks in some way to bring the learner model in line with the expert model. In this sense, the learner model is said to be an "overlay" of the expert model Wenger, 1987)

Although the various ITS research and development communities seem to agree that these are the main components, the model is really more of a conceptual framework than a working blueprint. In practice, different systems employ quite different architectures, data structures, and strategies, reflecting different instructional philosophies and purposes (e.g., Nkambou et al., 2010, Schatz & Folsom-Kovarik, 2011). While this makes sense locally, the lack of a standard overall system architecture and way of constructing the different system components is problematic for a number of reasons.

For one thing, it means that components that have proven to be effective in one system are not easily imported into another, thus limiting progress that might be made through the collective efforts of the rapidly expanding network of ITS research and development groups around the world. Also, the lack of a standard method of structuring the learner model means that when a learner moves from one system to the

[4] Although we refer here to "user" as a "learner," in fact the agent we describe here is capable of evaluating the discourse moves of any interlocutor, including those of another agent.

next, the new system must start from scratch in establishing the learner's history and current state of knowledge. To use an analogy from medicine, it is as if different doctors had different languages for describing a patient's health, so, when dealing with a new patient, each doctor would have to reassemble a patient's history from scratch.

What then, in the absence of a standard architecture and data structure for representing learner and expert domain knowledge, is one do to? The 60s-era slogan "If you're not part of the solution, you're part of the problem" seems relevant here. Specific solutions to general problems of learner modeling ought to be crafted in such a way that they are generally useful no matter what environment they are asked to work in. To return to the medical records analogy, if we are developing a new procedure for say, measuring pupil dilation or "knee-jerk" reflex response, so long as we have a standard way of reporting our results, we don't have to worry about how our "microrecord" fits into the overall structure of the patient's medical record, which can be assembled by someone else. In other words, we can be part of a solution without knowing exactly what that solution is.

The Argument for a Multiagent Architecture

The notion of an ITS as a multiagent system is not new. Of course, any ITS is a multiagent system in the gross sense that there are two autonomous agents at work: the user and intelligent tutor. However, recent years have seen an increasing emphasis on development of ITSs with multiagent architectures in the more interesting sense that overall system functionality emerges from the collective work of individual software agents (Bittencourt et al., 2007; Chen & Mizoguchi, 2004; El Mokhtar En-Naimi, Amami, Boukachour, Person & Bertelle, 2012; Lavendelis & Grundspenkis, 2009; Zouhair et al., 2012). Through the use of a shared, speech-act-based agent communication framework such as Knowledge Query and Manipulation Language (KQML) (Finin, Fritzson, McKay & McEntire, 1994), Foundation for Intelligent Physical Agents (FIPA)-ACL (O'Brien & Nicol, 1998), or Java Agent Development Framework (JADE) (Bellifemine, Caire, Poggi & Rimassa, 2008), combined with a set of domain-specific ontologies (concepts and their relations specific to the system domain), the agents in the system assert beliefs, make requests of other agents, deny requests, and so forth, much as human workers in a large collective enterprise do (see Chaib-draa & Dignum, 2002; Kone, Shimazu & Nakajima, 2000).

For example, in a multiagent ITS, different agents can take on the different tasks of user registration and authentication, interfacing with learning management systems, building learner models dynamically, monitoring learner affect through the use of various sensing systems, and managing conversations among users and other agents. Further, agents with especially complex tasks, such as a conversation agent, may be supported by a network of specialized agents dedicated to specific subtasks. Each of these agents can have its own internal algorithms, data structures, and local methods of obtaining data. Importantly, as long as the agent "knows" the system's ACL, i.e., can post, send, and read messages in a shared language, it doesn't matter how it is organized internally, in the same way that different sort functions can take the same input and produce the same output using different internal algorithms.

In the remainder of the chapter, we give an example: an agent that is capable of evaluating a learner's discourse moves on two important dimensions: *relevance* and *novelty*. Instantiated as a highly specialized agent within a multiagent, conversation-based ITS architecture, the R-N agent is capable of making assertions about a small but important piece of the learner model (a learner micromodel), information that may be of some value to other agents, such as conversation agents and learning model agents, for their own purposes.

The Problem of Conversational Relevance and Novelty

Designing a computer program that can carry on a conversation with a human, one of the oldest and arguably the hardest challenge in AI, is exactly the sort of problem that lends itself to multiagent treatments. For example, different agents may be responsible for converting speech to text, parsing text into its grammatical elements, classifying utterances as different kinds of speech acts, and extracting (or estimating) meaning through some form of semantic analysis. Other agents, or clusters of agents, can be responsible for generating responses in the form of text strings, while still other agents convert the text strings into speech, and have them spoken by an animated avatar.

The agent responsible for generating responses to a user's utterances (discourse moves) arguably has the most complex task and is most likely to benefit from the assistance of simpler agents that can take on pieces of it. The task is hard because it is really a special form of mind-reading, requiring an ability to continuously create and test theories about an interlocutor's present state of mind. Bakhtin's distinction between monologic and dialogic discourse (Bakhtin, 1981; Wells, 2007) is a useful way of framing the problem. In a conversation that is primarily monologic, the speaker's purpose is to convey information in such a way as to "duplicate one's own idea in someone else's mind" (Bakhtin, 1986:69), without the need to be concerned about what is already in the listener's mind, what the listener may be thinking at the present moment or in expectation of a particular response. This is the discourse stance of a lecturer. Dialogic discourse, on the other hand, is inherently a "social form of thinking" (Wells, 2007:256), a much harder form of discourse, at least for machines, in which interlocutors must work to understand each other's "present state," and thereby arrive at a shared understanding.[5] Consider, for example, the following exchange:

A: So, what did you do today?
B: Attended a physics lecture.
A: What did you learn?
B: It was really hot.

Given such a response, if A is a human speaker, A will assume, on the Gricean principle, that B is a cooperative interlocutor; that B's response must be in some way *relevant* (Grice, 1975); and that "it" refers either to the lecture or the lecture hall. In the former case, "hot" would be an attribute of the lecture, representing a positive reaction, implying that speaker B had liked the lecture and possibly learned a lot from it. In the latter case, "hot" would be an attribute of the lecture hall, implying an uncomfortable temperature. Because the word "hot" is more commonly associated with rooms than lectures, A would probably test the latter case first:

…
A: So, are you saying the lecture hall was uncomfortable?
B: Yes.
A: I'm sorry to hear that. It must have been hard to concentrate. What were you able to learn?
B: We learned about physics.
A: Okay, but what about physics?

[5] In a paper titled "Why is conversation so easy?" Garrod & Pickering (2004) argue that, for humans, dialog is easier than monologue because interlocutors automatically align linguistic representations at various levels (phonological, syntactic, semantic, and situational), thus building up a shared workspace. This joint construction of meaning and purpose has the effect of distributing the processing load, thus making conversation relatively easy. Humans, as they put it, are "designed for dialog."

In order for an intelligent conversation agent to carry on such a conversation, it would need, among many other things, to have some knowledge of the world, of the following type:

1. Lectures are a method of teaching.
2. Lectures are about something.
3. A person may learn something from a lecture.
4. Lectures take place in rooms called lecture halls.
5. Rooms may be comfortable or uncomfortable.
6. Humans are sensitive to temperatures that are outside their comfort range.
7. “Hot” refers to a temperature that is outside a human’s comfort range.
8. Learning requires concentration.
9. When a person is uncomfortable, it is hard to learn.
10. “Hot” is a slang word for something that a human finds attractive…

and so forth. In other words, A’s ability to judge the *relevance* of B’s utterance “It was very hot” depends very much on a complex set of concepts and relationships.

However, relevance is not the only important measure of the degree to which a discourse move is felt to be cooperative. In addition to the principle of relevance is that of *quality* (Grice, 1975), which includes the assumption that an interlocutor’s move will add something *new* to the conversation. In the imagined conversation considered here…

Learner (B): We learned about physics.
Tutor (A): Okay, but what *about* physics?

A’s “Okay, but what *about* physics?” is exactly the right thing to say because B has already reported that it was a *physics* lecture, so of course it was about physics. In other words, B has violated the Gricean maxim of quality by failing to make a truly novel contribution.

So, if it is to possess anything remotely like the intelligence of a human speaker, it seems that an intelligent conversation agent must have a way of evaluating both the relevance and novelty of an interlocutor’s discourse moves. Admittedly, this is just one part of the problem of natural language understanding by a machine, perhaps even a minor part, and so a solution, which itself is likely to be only partial, will be worth little if not combined with many others. That said, within a multiagent environment, it might be useful to have an agent that is capable of sending messages like:

```
json={“target”:”conversation manager”,”learner”:”5021”,”time”:”2013-02-01
10:45:22 UTC”, “novelty”:” .3”,”relevance”:”.5”}
```

or

```
json={“target”:”learner model manager”,”learner”:”1259”,
“session”:”5”,”date”: “2013-02-01”,”average novelty”:”medium”, “average
relevance”:”high”}
```

where the language is an ACL, the decimals represent Bayesian estimates of likelihood, and the assertions are addressed to other agents with a “for your information” (FYI) intention, i.e., the originating agent is agnostic as to how other agents will use the information (beliefs) it asserts.[6]

An Example from Physics

As an example, assume that an intelligent tutor is provided with the following instructional script:

ASK: *Suppose a football player is running in a straight line at a constant speed, while carrying a football. Suddenly he throws the football straight up, over his head, and keeps running at the same speed. Explain where it will land.*

EXPECTED ANSWER: *In accordance with Newton’s first law of motion, the ball will continue to travel in the same direction, unless there is an opposing force. In this case, the main opposing force is gravity, but gravity only affects the vertical component of the motion vector. It is true that the particles in the air, such as oxygen molecules, exert an opposing force, which pushes against the ball in the opposite direction, but this force is very small compared to the accelerating force of gravity.*

Now, compare the following two transcripts from an imaginary tutoring session:

Transcript A

Tutor: Suppose a football player is running in a straight line at a constant speed, while carrying a football. Suddenly he throws the football straight up, over his head, and keeps running at the same speed. Explain where it will land.
Learner A: It will continue along in his direction and come back down where he can catch it.
Tutor: I agree. But why?
Learner A: Well, the ball and the runner both have the same forward speed.
Tutor: Okay...and....
Learner A: So when gravity pulls the ball down, it just affects the vertical speed.
Tutor: What about the horizontal speed?
Learner A: It remains the same. It’s Newton’s second law of motion.
Tutor: Do you mean Newton’s first law?
Learner A: Oh right. First law.
Tutor: Good. Let’s go on to the next problem. What if, instead of a ball, it’s a balloon? On the moon?

Transcript B

Tutor: Suppose a football player is running in a straight line at a constant speed, while carrying a football. Suddenly he throws the football straight up, over his head, and keeps running at the same speed. Explain where it will land.

Learner B: i don’t know. somewhere behind him.
Tutor: Why?
Learner B: cause it will fall back behind him so he can’t catch it.

[6] The syntax used here is JavaScript Object Notation (json), a message format used for communication between agents in multiagent systems. See http://www.json.org/.

Tutor: Why do you think that?
Learner B: i just think it will fall behind him
Tutor: Okay...but why?
Learner B: cause
Tutor: Okay, let's look at that video again....

Clearly, there is quite a lot going on here; however, a key difference between the two transcripts is that whereas Learner A's contributions are all *relevant* (consistent with the expected contribution) and *novel* (providing additional information); in the same sense, only Learner B's first contribution is novel, and none are relevant.[7] More generally, the relevance and novelty of given discourse move may be mapped onto a Venn diagram such as that shown in

Figure 9-2.

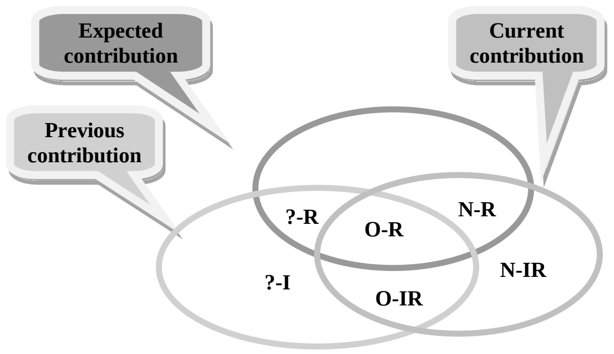


Figure 9-2. Mapping relevance and novelty

This gives a matrix (Table 9-1) with four quadrants:

Table 9-1. Novelty and Relevance Matrix

	Old	**New**
Relevant	O-R (Not new but relevant)	N-R (New and relevant)
Irrelevant	O-IR (Not new and irrelevant)	N-IR (New but irrelevant)

Assuming that this kind of information would be useful to other agents (notably the agent handling the conversation with the learner), we can now ask how an agent might go about determining the relevance and novelty of a given utterance.

[7] Note that we are using the term "relevant" here in a special, non-intuitive sense. Whereas a contribution may be "relevant" in the sense that it relates in some way to the topic, it is considered irrelevant if it is inconsistent with a model answer.

Quantitative Measures for Novelty and Relevance

A rough-and-ready relevance measure for a given learner contribution to a dialog with an intelligent tutor can be defined as the extent to which the learner's answer to a tutor's question is "semantically similar" to the answer the tutor expects. In the same way, a measure of the novelty of the learner's most recent contribution can be defined as the extent to it resembles the learner's previous contributions to the same conversation, i.e., attempts to answer the same question. In other words, if the R-N agent is passed two strings – one representing the expected answer and the other the learner's most recent contribution – then it can it can come up with a relevance score using any one of several methods used to compute semantic similarity, including Latent Semantic Analysis (LSA; Landauer & Dumais, 1997), Hyperspace Analogue to Language (HAL; Burgess, Livesay & Lund, 1996), Latent Dirichlet Allocation (LDA; Blei, Ng & Jordan, 2003), Non-Latent Similarity (NLS; Cai et al., 2004); Word Association Space (WAS; Steyvers, Shiffrin & Nelson, 2002), and Pointwise Mutual Information (PMI; Recchia & Jones, 2009). For a discussion of the use of LSA in ITSs see Hu et al. (2007).

Also, so long as it knows that the topic has not changed (e.g., the tutor is still prompting for an answer to the same question), then, using the same method, it can calculate the semantic similarity of the learner's most recent contribution to her previous contributions. This produces values seens in Table 9-2.

Table 9-2. Sample relevance and novelty measures

	Old	**New**
Relevant	0.4 (O-R)	0.2 (N-R)
Irrelevant	0.1 (O-IR)	0.3 (N-IR)

In addition, two other measures are obtained by combining current and previous contributions. A Current Relevant Contribution (CRC) score is obtained by adding O-R and N-R (in this case, 0.6), while the CRC score combined with all previous CRC scores gives a Total Coverage (TC) score.

Together, for any given dialog move, these six measures may be viewed as constituting a micromodel of the learner's "current state." An agent that is capable of evaluating a given dialog move on these measures can pass along the micromodel it has built for the use of other agents in the community. For example, in Transcript A, an agent's analysis of Learner A's contribution "Well, the ball and the runner both have the same forward speed" might be communicated as follows:

```
json={"target":"all","learner":"1259","input string":"Well, the ball and the
runner both have the same forward speed","time":"2013-02-05 11:25:27
UTC","R/N":"0.3","R/O":"0.4", "I/N":"0.28","I/O":"0.01",
"CRC":"0.17","TC":"0.24"}
```

...whereas Learner B's contribution "cause" could yield the following:

```
json={"target":"all","learner":"1147","input string":"cause","time":"2013-02-
05 11:25:27 UTC","R/N":"0.0",
"R/O":"0.0","I/N":"0.0","I/O":"0.10","CRC":"0.04","TC":"0.033"}
```

Over a series of moves, the cumulative scores constitute what Hu & Martindale (2008) refer to as a Learner Characteristic Curve (LCC), which may be displayed in the form of a set of graphs, as illustrated in Figure 9-3.

This information, the relevance and novelty of a given utterance, combined with the cumulative relevance measures, can be viewed as constituting a small, localized micromodel of the learner's current cognitive state and as such has practical utility. In fact, it is on the same level as micromodels developed by other agents in a multiagent ITS community that provides both real-time and historical information about a user's apparent affect.

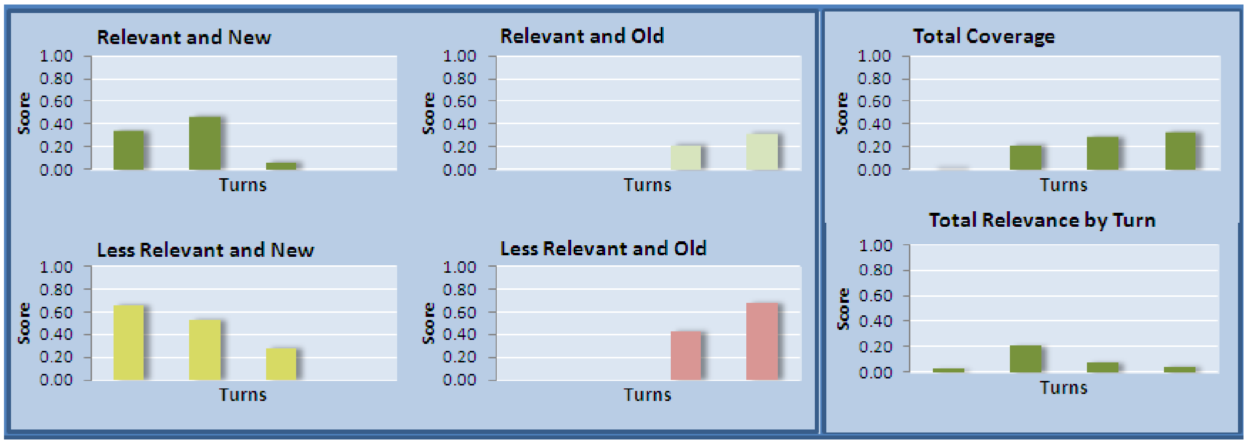


Figure 9-3. Sample LCC output

For example, an agent that monitors a learner's facial expressions might make an assertion in the form:

```
json={"target":"all","learner":"1147","facial
expression":"puzzled","value":".3","time":"2013-02-01 10:45:22 UTC"}
```

which constitutes a component micromodel of the learner's affective state. Combined with the preceding message from the R-N agent, the conversation manager (more specifically, a response generation agent) now has two pieces of evidence to consider, i.e., that the user appears confused and that the novelty and relevance measures for the user's most recent utterance were both low. Given this information, the response agent could decide to send a message such as this to an avatar:

```
json={"target":"avatar","learner":"1147","output string":"You seem confused.
Are you?","time":"2013-02-01 10:45:24"}
```

Assuming they have ways of understanding messages such as these, other agents in the system can use them for their own purposes. For example, a response generation agent might use the information in the first message to generate the turn:

Learner A: Well, the ball and the runner both have the same forward speed.
Tutor: Okay...and....

and the information in the second message to produce:

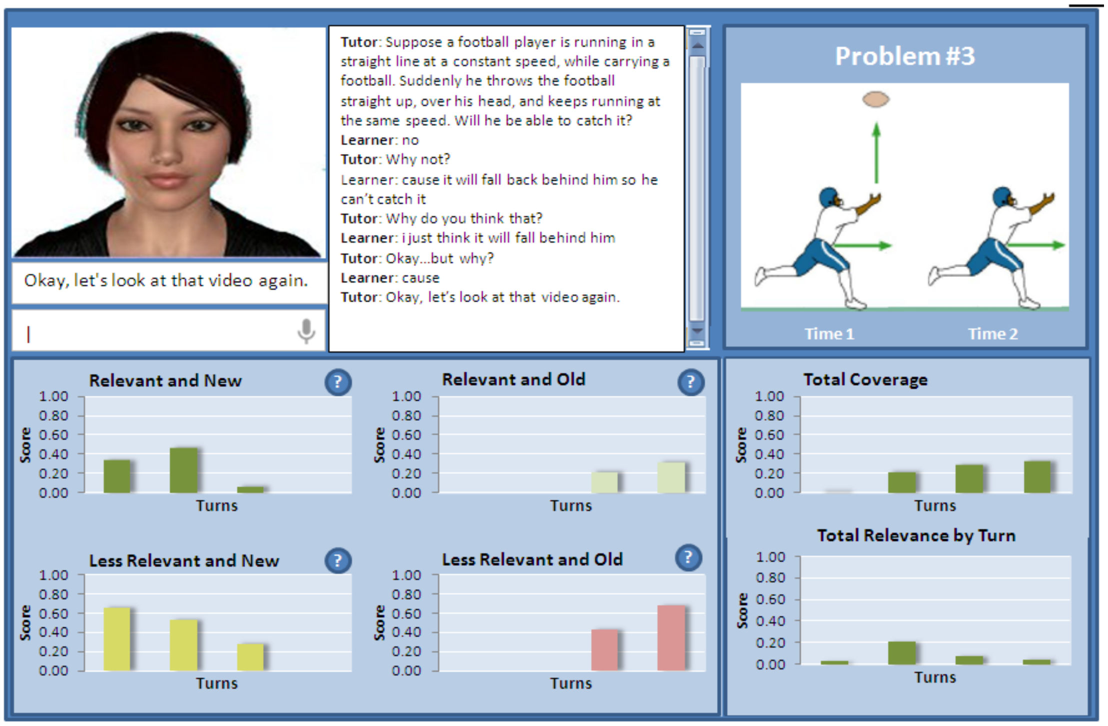
Learner B: cause
Tutor: Okay, let's look at that video again....

As another example, a pedagogical agent might take an LCC representing repeated non-novel, "irrelevant" learner contributions as evidence of a possible misconception.

Opening the Model to the Learner

In a system where the learner model is "open" to the learner (Bull, 2004; Bull & Pain, 1995; Dimitrova et al, 2001; Kay, 1997; Mitrovic & Martin, 2002), a user interface agent might use the micromodel to create a set of graphs, as in Figure 9-4.

.



Okay, let's look at that video again.

Tutor: Suppose a football player is running in a straight line at a constant speed, while carrying a football. Suddenly he throws the football straight up, over his head, and keeps running at the same speed. Will he be able to catch it?
Learner: no
Tutor: Why not?
Learner: cause it will fall back behind him so he can't catch it
Tutor: Why do you think that?
Learner: i just think it will fall behind him
Tutor: Okay...but why?
Learner: cause
Tutor: Okay, let's look at that video again.

Figure 9-4. Opening the micromodel to the learner

Giving the learner feedback on the relevance and novelty of her discourse moves in this way could conceivably encourage her to focus her attention on maintaining higher levels of relevance and novelty than she might otherwise, thereby increasing the likelihood that the conversation will lead to real learning.

Discussion

In this chapter, we have explained how a specialized agent within a conversation-based ITS can monitor a learner's discourse moves on two dimensions: novelty and relevance. In this way, over a series of moves, it builds up a "micromodel" of the learner's cognitive state, called a Learner Characteristic Curve (LCC), which it can then pass along in the form of messages (assertions of belief) to other agents, assuming a common ACL. As a result, the agent can contribute to the solutions of larger problems without needing to know what the solution is. Importantly, such an agent is both reusable and replaceable. It is reusable in the sense that it can be used in any number of different ITSs where measures of novelty and relevance are considered useful in some way. It is replaceable in the sense that another agent that performs the same analysis, but more effectively, could be brought in to take over. This form of loose coupling (Orton & Weick, 1990; Weick, 1976) allows for the rise of mutations at both the local (agent) and system (multiagent) levels, thus allowing gradual evolution toward increasingly sophisticated systems that may eventually approach the effectiveness of a highly skilled human teacher.

What, then, are the implications of the preceding argument for a "generalized intelligent framework" like GIFT? More precisely, how can we, as a community of ITS researchers and developers, possibly move forward from our current world of "domain-specific, unique, one-of-a-kind, largely domain-dependent solutions focused on a single pedagogical strategy" (Sottilare, Brawner, Goldberg & Holden, 2012:1) toward a future of open, domain-independent systems with shareable, reusable tools and components, efficient authoring, transportable learner models, cross-platform functionality, and so forth. Given our rapidly evolving technological environment (e.g., the sudden ubiquity of smartphones and tablets, the explosion of massive, text-based knowledge representations in the Semantic Web, the magnetic attraction of social media such as Facebook and YouTube, etc.), it seems unlikely that a single, intelligent tutoring solution will ever be more than temporarily useful. Rather, it seems what we can look forward to, and should build toward as a community of practice, is some evolving collection of workable, partial, and loosely coupled solutions, which, while provisional, are built in such a way that they can evolve both with and apart from each other. Specifically, the recommendation is that we begin to think seriously about the adoption of a common ACL such as KQML (Finin, Fritzson, McKay & McEntire, 1994), FIPA-ACL (O'Brien & Nicol, 1998), or JADE (Bellifemine, Caire, Poggi & Rimassa, 2008) as the basis for a new generation of agent-based intelligent learning systems that are capable of *autonomous cooperation* (Hülsmann, Scholz-Reiter, Freitag, Wucisk & De Beer, 2006; Windt, Böse & Philipp, 2005). Importantly, because the contributing agents can have their own internal databases and algorithms, we can move in this direction without wholesale reengineering of our existing systems. Rather, once we agree on a common ACL and begin to build a shared ontology (which can occur incrementally), then we can "simply" encapsulate our existing (and evolving) systems in wrappers that allow these systems, whatever their function, to take part in whatever communities may arise, both adding and receiving value, in ways that are now only dimly imaginable.

References

Allen, J. E., Guinn, C. I. & Horvtz, E. (1999). Mixed-initiative interaction. *Intelligent Systems and their Applications, IEEE, 14*(5), 14-23.

Bakhtin, M. M. [1930s] 1981. *The dialogic imagination: Four essays*. Ed. Michael Holquist. Trans. Caryl Emerson and Michael Holquist. Austin and London: University of Texas Press.

Bellifemine, F., Caire, G., Poggi, A. & Rimassa, G. (2008). JADE: A software framework for developing multiagent applications. *Lessons learned. Information and Software Technology, 50*(1), 10-21.

Bittencourt, I. I., de Barros Costa, E., Almeida, H. D., Fonseca, B., Maia, G., Calado, I. & Silva, A. D. (2007). Towards an ontology-based framework for building multiagent intelligent tutoring systems. In *Workshop on software engineering for agent-oriented systems, III, João Pessoa, 2007*. Porto Alegre, SBC, 53-64.

Blei, D. M., Ng, A. Y. & Jordan, M. I. (2003). Latent dirichlet allocation. *The journal of machine learning research, 3*, 993-1022.

Bull, S. & Pain, H.(1995). Did I say what I think I said, and do you agree with me? Inspecting and questioning the student model. In J. Greer (Ed.), *Proceedings of World Conference on Artificial Intelligence in Education, Association for the Advancement of Computing in Education*, 501-508

Bull, S. (2004). Supporting learning with open learner models. *Planning, 29*(14), 1.

Burgess, C. & Lund, K. (1997). Modeling parsing constraints with high-dimensional context space. *Language and Cognitive Processes , 12*, 177-210.

Cai, Z., McNamara, D. S., Louwerse, M., Hu, X., Rowe, M. & Graesser, A. C. (2004). Nls: Non-latent similarity algorithm. In K. Forbus, D. Gentner & T. Regier (Eds.), *Proceedings of the 26th annual meeting of the cognitive science society* (p. 180-185). Mahwah, NJ:Erlbaum.

Carbonell, J. R. (1970). AI in CAI: Artificial intelligence approach to computer assisted instruction. *IEEE Transactions on Man-Machine Systems 11(4)*: 190-202.

Chaib-draa, B. & Dignum, F. (2002). Trends in agent communication language. *Computational Intelligence,* 18(2), 89-101.

Chen, W. & Mizoguchi, R. (2004). Learner model ontology and learner model agent. *Cognitive Support for Learning-Imagining the Unknown*, 189-200.

Corbett, A. T., Koedinger, K. R. & Anderson, J. R. (1997). Intelligent tutoring systems. *Handbook of human-computer interaction*, 849-874.

El Mokhtar En-Naimi, A. Z., Amami, B., Boukachour, H., Person, P. & Bertelle, C. (2012). Intelligent Tutoring Systems Based on the Multiagent Systems (ITS-MAS): The Dynamic and Incremental Case-Based Reasoning (DICBR) Paradigm. *IJCSI International Journal of Computer Science Issues 9*(6), 112-121.

Elson-Cook, M. (1993). Student modelling in intelligent systems. *Artificial intelligent review, 7*(3-4), 227-240.

Finin, T., Fritzson, R., McKay, D. & McEntire, R. (1994, November). KQML as an agent communication language. In *Proceedings of the third international conference on Information and knowledge management* (pp. 456-463). ACM.

Garrod, S. & Pickering, M. J. (2004). Why is conversation so easy?*Trends in cognitive sciences, 8*(1), 8-11.

Graesser, A. C., Conley, M. W. & Olney, A. (2012). Intelligent tutoring systems. *APA handbook of educational psychology*. Washington, DC: American Psychological Association.

Graesser, A. C., Jeon, M. & Dufty, D. (2008). Agent technologies designed to facilitate interactive knowledge construction. *Discourse processes, 45*, 298–322.

Graesser, A. C., Olney, A. M., Haynes, B. C. & Chipman, P. (2005). AutoTutor: A cognitive system that simulates a tutor that facilitates learning through mixed-initiative dialogue. In C. Forsythe, M. L. Bernard & T. E. Goldsmith (Eds.), *Cognitive systems: Human cognitive models in systems design*. Mahwah, NJ: Lawrence Erlbaum.

Graesser, A.C., Lu, S., Jackson, G.T., Mitchell, H., Ventura, M., Olney, A. & Louwerse, M.M. (2004). AutoTutor: A tutor with dialogue in natural language. Behavioral research methods, instruments, and computers, 36, 180-193.

Grice, H. Paul. "Logic and conversation. Speech acts, ed. by Peter Cole and Jerry Morgan, 41-58." (1975).

Hu, X., Cai, Z., Wiemer-Hastings, P., Graesser, A. C. & McNamara, D. S. (2007). Strengths, limitations, and extensions of LSA. *The handbook of latent semantic analysis*, 401-426.

Hu, X. & Martindale, T. (2008, January). Enhance learning with ITS style interactions between learner and content. In The Interservice/Industry Training, Simulation & Education Conference (I/ITSEC) (Vol. 2008, No. -1). National Training Systems Association.

Hülsmann, M., Scholz-Reiter, B., Freitag, M., Wucisk, C. & De Beer, C. (2006). Autonomous cooperation as a method to cope with complexity and dynamics?–A simulation based analyses and measurement concept approach. In Y. Bar-Yam (Ed.), *Proceedings of the International Conference on Complex Systems* (ICCS 2006). Boston, MA, USA (Vol. 2006).

Kay, J. (2001). Learner control. *User Modeling and User-Adapted Interaction, 11*(1), 111-127.

Kone, M. T., Shimazu, A. & Nakajima, T. (2000). The state of the art in agent communication languages. *Knowledge and Information Systems, 2*(3), 259-284.

Landauer, T. K. & Dumais, S. T. (1997). A solution to Plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge.*Psychological Review; Psychological Review, 104*(2), 211.

Lavendelis, E. & Grundspenkis, J. (2009). Design of multiagent based intelligent tutoring systems. *Scientific Journal of Riga Technical University. Computer Sciences, 38*(38), 48-59.

Mitrovic, A., Martin, B. & Suraweera, P. (2007) Intelligent tutors for all: Constraint-based modeling methodology, systems and authoring. *IEEE Intelligent Systems, 22*(4), 38-45.

Nkambou, R., Mizoguchi, R. & Bourdeau, J. (2010). *Advances in intelligent tutoring systems*. Heidelberg: Springer.

O'Brien, P. D. & Nicol, R. C. (1998). FIPA—towards a standard for software agents. *BT Technology Journal, 16*(3), 51-59.

Orton, J. D. & Weick, K. E. (1990). Loosely coupled systems: A reconceptualization. *Academy of management review*, 203-223.

Person, N. K., Bautista, L., Kreuz, R. J., Graesser, A. C. & Tutoring Research Group. (2000). The dialog advancer network: A conversation manager for AutoTutor. In *ITS 2000 Proceedings of the Workshop on Modeling Human Teaching Tactics and Strategies* (pp. 86-92).

Picard, R. (2006). Building an affective learning companion. Keynote address at the *8th International Conference on Intelligent Tutoring Systems*, Jhongli, Taiwan.

Psotka, J., Massey, L. D. & Mutter, S. A. (Eds.). (1988). *Intelligent tutoring systems: Lessons learned*. Lawrence Erlbaum.

Recchia, G. & Jones, M. N. (2009). More data trumps smarter algorithms: Comparing pointwise mutual information with latent semantic analysis. *Behavior research methods, 41*(3), 647-656.

Ritter, S., Anderson, J. R., Koedinger, K. R. & Corbett, A. (2007). Cognitive Tutor: Applied research in mathematics education. *Psychonomic bulletin & review, 14*(2), 249-255.

Schatz, S. & Folsom-Kovarik, J. T.(Winter, 2011). Return on Investment: A practical review of learner modeling techniques. *M&S Journal*. www.msco.mil

Sleeman D. & J. S. Brown. (1982). (Eds.). *Intelligent tutoring systems*. Orlando, Florida: Academic Press.

Sottilare, R.A., Brawner, K.W., Goldberg, B.S. & Holden, H.K. (2012). *The Generalized Intelligent Framework for Tutoring (GIFT)*. Orlando, FL: U.S. Army Research Laboratory – Human Research & Engineering Directorate (ARL-HRED).

Steyvers, M., Shiffrin, R. M. & Nelson, D. L. (2002). Semantic spaces based on free association that predict memory performance. *Triple Festschrift honoring Lyle Bourne, Walter Kintsch, and Tom Landauer.*

VanLehn, K. (2006). The behavior of tutoring systems. *International journal of artificial intelligence in education, 16*(3), 227-265.

Weick, K. E. (1976) Educational organizations as loosely coupled systems. *Administrative Science Quarterly, 21*, 1-19.

Wells, G. (2007. Semiotic mediation, dialogue and the construction of knowledge. *Human Development*. 50(5). pp 244-274.

Wenger, E. (1987). *Artificial intelligence and tutoring systems—Computational and Cognitive Approaches to the Communication of Knowledge*. Los Altos, CA: Morgan Kaufmann.

Windt, K., Böse, F. & Philipp, T. (2005). Criteria and application of autonomous cooperating logistic processes. In *Proceedings of the 3rd International Conference on Manufacturing Research. Advances in Manufacturing Technology and Management*. Gao, JX and Baxter, DI and Sackett, PJ (Eds.).

Woolf, B. P. (2008). *Building intelligent interactive tutors: Student-centered strategies for revolutionizing e-learning*. Morgan Kaufmann.

Wolfe, C. R., Fisher, C. R., Reyna, V. F. & Hu, X. (2012). Improving internal consistency in conditional probability estimation with an Intelligent Tutoring System and web-based tutorials. *International Journal of Internet Science*, 7, 38-54.

Zouhair, A., En-Naimi, E. M., Amami, B., Boukachour, H., Person, P. & Bertelle, C. (2012, October). Intelligent tutoring systems founded on the multiagent incremental dynamic case based reasoning. In *Information Science and Technology (CIST), 2012 Colloquium* (pp. 74-79). IEEE.

CHAPTER 10 –Learner Models in the Large-Scale Cognitive Modeling (LSCM) Initiative

Scott A. Douglass
U.S. Air Force Research Laboratory (AFRL) - Cognitive Models and Agents Branch (RHAC)

Introduction

AFRL research efforts seeking to exploit cognitive modeling are growing in scale and complexity. These efforts are struggling to meet the challenges of increasing the scale of cognitive models and integrating them into software-intensive training environments. The struggle has two sources: (1) the need to specify detailed knowledge and process descriptions in currently available modeling frameworks; and (2) a dependence on specialized simulators in contemporary cognitive modeling frameworks that isolate models from standards, methods, and tools used by the larger systems engineering community. As long as the struggle remains, AFRL faces capability gaps preventing cognitive science and cognitive modeling from having maximal impact on training and human effectiveness.

For the last three years, an AFRL LSCM research initiative has sought to close capability gaps retarding the development and fielding of intelligent training systems technologically based on cognitive models and agents. Generally, the LSCM initiative has endeavored to enhance and accelerate the practice of cognitive modeling. Specifically, the LSCM initiative has researched and developed the following:

1. Domain-specific languages (DSLs) tailored to the needs of cognitive modelers.

2. Authoring environments in which users employ these DSLs to specify models and agents that are "correct by construction."

3. Code-generation technologies that transform models and agents specified in these authoring environments into executable artifacts.

4. A cognitively enhanced complex event processing (CECEP) architecture in which models and agents are executed.

5. A massively concurrent net-centric associative memory application (Douglass & Myers, 2010) enabling models and agents executing in the CECEP architecture to store and remember vast quantities of declarative knowledge.

6. A constraint-based knowledge representation and mining application (Douglass & Mittal, 2013) that allows models and agents executing in the CECEP architecture to produce and recognize actions using domain knowledge.

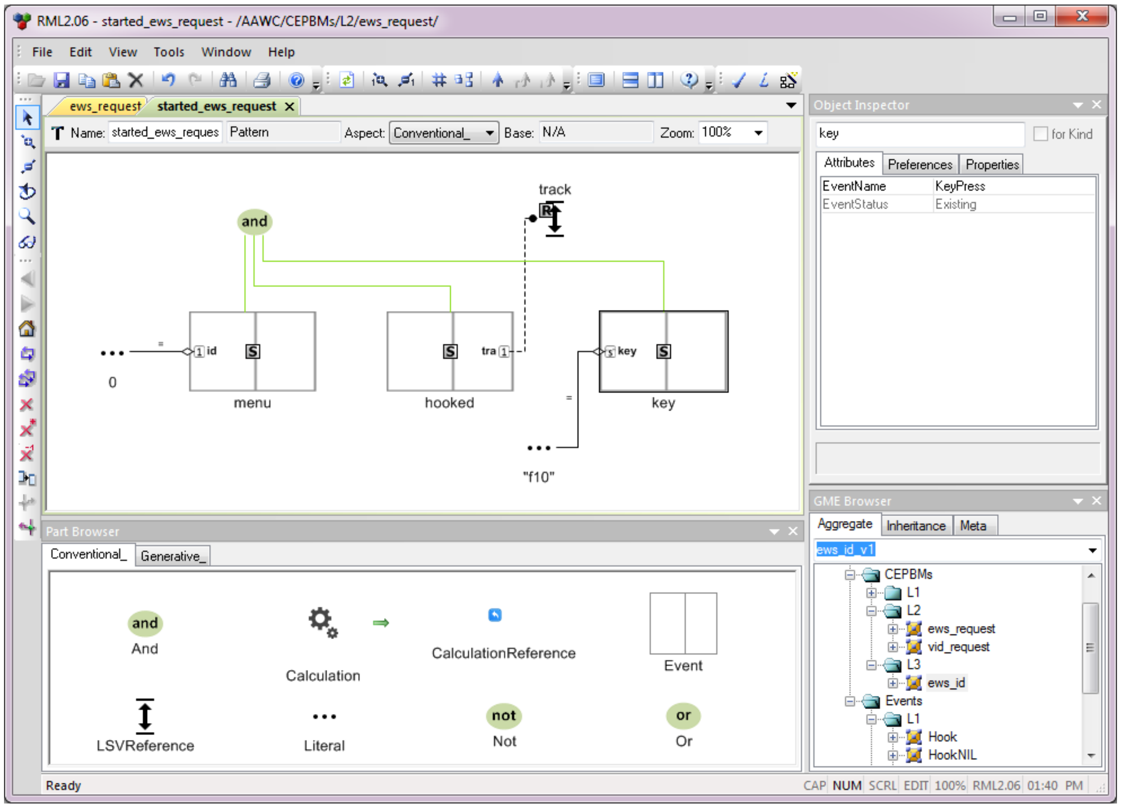
7. Cognitive model behavior analysis and visualization capabilities (Bogart et al., 2011) that help users understand and analyze models functioning more like autonomous agents than programs.

This chapter explains how learner models are represented and processed by instructional agents executing in the CECEP architecture. The chapter uses three sections to achieve this objective. The first section describes cognitive modeling and simulation in the LSCM initiative. This section introduces the reader to a LSCM DSL and the CECEP architecture in which models and agents specified in this DSL are executed. The second section extends the content of the first section to demonstrate how LSCM instructional agents track and make sense of learner actions. This section explains how LSCM instructional agents use behavior models (representations of procedural knowledge) and cognitive domain ontologies (representations of domain knowledge) to trace learner actions. The third section of the chapter explains how behavior models and cognitive domain ontologies constitute a learner model. This section describes how LSCM instructional agents use event monitoring based on behavior models and hypotheses about learner behaviors based on domain knowledge to evaluate the adequacy of learner actions, track skill acquisition, and predict future performance.

Cognitive Modeling and Simulation in the LSMC Initiative

Modeling in the LSCM Initiative

To decrease the time and cost of model and agent development in the AFRL, the LSCM research initiative is developing a domain-specific language DSL called the research modeling language (RML). The RML DSL is used by AFRL research scientists to specify cognitive models and agents that execute in a net-centric CECEP architecture. The RML DSL has been developed using the Generic Modeling Environment (GME), a meta-modeling tool for creating and refining domain-specific modeling languages and program synthesis environments (Ledeczi et al., 2001; Molnár et al., 2007). The abstract syntax of RML is influenced by the ACT-R cognitive architecture (Anderson, 2007). The graphical/textual concrete syntax of RML is designed so that modelers with experience in ACT-R, or domain experts with little modeling and programming experience, can specify models at a high level of abstraction. When the GME application is configured to "interpret" RML, it is transformed into the authoring environment shown in Figure 10-1.


RML2.06 - started_ews_request - /AAWC/CEPBMs/L2/ews_request/

Name: started_ews_reques Pattern Aspect: Conventional_ Base: N/A Zoom: 100%

Object Inspector: key

EventName	KeyPress
EventStatus	Existing

GME Browser: ews_id_v1

Figure 10-1. The RML agent authoring environment.

The authoring environment consists primarily of four interface panels: (1) an "Object Inspector" that can be used to inspect or edit textual attributes of RML specifications: (2) a "GME Browser" that can be used to organize and share repositories of RML specifications; (3) a "Part Browser" presenting contextually appropriate RML constructs to the user; and (4) tabbed work spaces in which users construct RML specifications. The RML authoring environment is based on a hybrid graphical/textual concrete syntax to support domain and subject matter experts rather than programmers. Constraints included in the formal abstract syntax of RML are used at runtime by GME to prohibit users from specifying invalid models and agents. If a models or agent can be specified in the authoring environment, they will produce "correct" executable code artifacts.

The RML DSL is designed to facilitate the specification of models and agents. In order to maximize scalability and interoperability during execution/simulation, RML and the GME-based authoring environment require users to conceive of and specify their models and agents as *complex event processing agents* (Luckham, 2002). The graphical/textual DSL allows cognitive modelers to rapidly specify models and agents in this way using representations of the following:

- *Events*: Events are objects that serve as records of activities in a system. Objects capture the details of events with attributes and data properties. In the context of an ITS, event objects represent trainee actions (user interface events, trainee eye movements, and trainee vocalizations, for example) tracked by instructional agents during performance monitoring.

- *Event Patterns*: Event patterns are templates matching one or more events in an event cloud constituting a representation of context functionally equivalent to a working memory. In the context of an ITS, event patterns enable instructional agents to detect correlational, temporal, and causal relationships between events representing activities in the training environment.

- *Event Pattern Rules*: Event pattern rules are associations specifying actions that are to occur after an event pattern is matched with a subset of events in the event cloud. In the context of an ITS, event pattern rules are used by instructional agents to produce abstract events that combine and aggregate attributes of event patterns. Abstract events are used to monitor trainee actions occurring at multiple levels during hierarchical tasks.

- *Behavior Models*: Bahvior models are sets of event pattern rules arranged into automata that explicitly represent behavior-specific combinations of cognitive state, contextual factors, alternative courses of action, and failure. In the context of an ITS, behavior models are used by instructional agents to monitor simple, complex, concurrent, and hierarchically organized trainee activities.

- *Cognitive Domain Ontologies*: Cognitive doman ontologies (CDOs) are representations of domain knowledge capturing: (1) entities, structures, and hierarchies in a domain; and (2) relations between these entities. In the context of an ITS, CDOs represent constraint knowledge that can be mined by and instructional agent to assess the intent and effectiveness of trainee actions.

Modeling in RML is based on the specification of procedural knowledge (behavior models) and domain knowledge (CDOs). Modelers do not have to attend to the details of how models and agents technically execute in simulations.

Simulation in the LSCM Initiative

To increase performance and maximize interoperability in training environments based on service-oriented architectures (SOAs), the CECEP architecture has been developed to serve as the execution framework for RML models and agents. The CECEP architecture consists of the following central net-centric components:

- *soaDM*: an associative memory application that allows RML models and agents to store and retrieve *declarative knowledge*. Declarative knowledge is represented and processed in a semantic network (Douglass & Myers, 2010).

- *soaCDO*: a knowledge representation and mining application that allows RML models and agents to store and exploit *domain knowledge*. Domain knowledge is represented in CDOs, which are processed by a constraint-satisfaction framework (Douglass & Mittal, 2013).

- *Esper*: a complex event processing framework that allows RML models and agents to base actions on context assessment and *procedural knowledge*. Procedural knowledge is represented in RML behavior models and processed using pattern matching and event abstraction capabilities provided by Esper (http://esper.codehaus.org/).

Through these components, the CECEP architecture incorporates model and agent capabilities based on declarative, procedural, and domain knowledge processing to the Esper framework. The resulting event-driven architecture is an advanced cognitive modeling and simulation framework with which AFRL research scientists can develop and field performance assessment and instructional technologies based on cognitive models and agents. A functional representation of the CECEP architecture is shown in Figure 10-2.

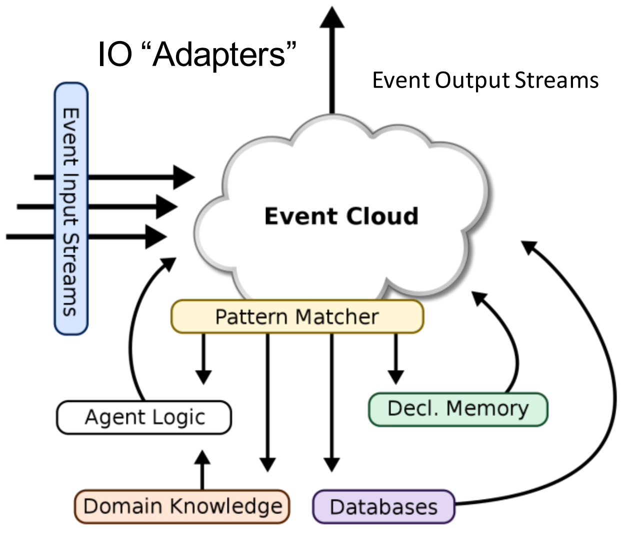


Figure 10-2. The CECEP architecture in which RML models and agents are executed

The CECEP architecture includes a number of input/output (IO) "Adapters" or event IO streams. These adapters allow models and agents specified in RML to be integrated into software-based instructional systems. The architecture includes event sources based on soaDM, soaCDO, and Esper. RML behavior models interacting with Esper enable agent logic. CDOs processed in soaCDO enable domain knowledge.

Finally, semantic networks processed in soaDM enable declarative memory. Figure 10-2 illustrates with arrows how these event sources produce events through: (1) the execution of agent logic; (2) the querying of databases containing long-term knowledge; (3) the mining of domain knowledge; and (4) interactions with a large-scale associative or declarative memory capable of mimicking the ACT-R cognitive architecture's activation-based declarative memory. It is these event sources based on agent logic, declarative memory, domain knowledge, and databases that "cognitively enhance" CEP in CECEP. The event cloud (a form of working memory) and pattern matcher (a form of rule engine) in the architecture are technologically realized through Esper.

Models and agents specified in the RML authoring environment are not directly executed. RML specifications are instead translated into executable code artifacts in the following ways:

- *Declarative Knowledge* is specified as events and relations and processed by code generators that produce files that configure the soaDM semantic network.

- *Procedural Knowledge* is specified as behavior models and processed by code generators that produce NERML, a text-only DSL formally equivalent to RML that is then translated into Java (Hansen & Douglass, submitted). Code generation is divided into these steps to support modelers that prefer to specify models and agents directly in NERML. Java files generated from either RML or NERML interact with Esper and govern the behavior of models and agents.

- *Domain Knowledge* is specified in CDOs and processed by code generators that produce constraint-networks for use in soaCDO.

Esper is capable of monitoring in excess of 500,000 events per second and therefore allows the CECEP architecture to scale. The architecture's IO Adapters interface greatly simplifies integration and interoperability with services in a broader SOA-based training and operational systems; new capabilities and training environments are integrated into CECEP through a new adapter and shared events.

Tracking and Recognizing Learner Activity with RML

In this section, learner models are discussed in the context of an anti-air warfare coordinator (AAWC) task described by Anderson et al. (2004). Fu et al. (2006) comprehensively describe the AAWC task requirements and a model-based AAWC training system used to instruct human trainees and ACT-R trainee models. The tutor discussed in Fu et al. (2006) instructed trainees learning how to relate information cues describing aircraft (altitude, speed, type of radar, etc.) in a monitored airspace to categories of intentions (hostile, friendly, etc.) and aircraft types (commercial, strike, etc.). Trainees were instructed to investigate and classify as many aircraft as possible during dynamic 6-min scenarios.

To assess the usability of RML, its developers used it to build an instructional agent replicating all of the functionality of the AAWC tutor described in Fu et al. (2006). The RML AAWC instructional agent is based on events, event patterns, event pattern rules, and behavior models. The agent additionally exploits domain knowledge specified in CDOs. This section of the chapter describes examples of the events, event patterns, event pattern rules, and behavior models making up the AAWC instructional agent. This section then introduces CDOs and illustrates how they can capture knowledge about the AAWC task domain. This section discusses these aspects of an RML instructional agent in order to illustrate how learner actions are tracked and assessed across events, behavior models, and CDOs in the CECEP architecture.

Composing Activities with Event Abstraction Hierarchies

The RML AAWC instructional agent functions in an *event abstraction hierarchy*. Figure 10-3 illustrates the portion of this hierarchy relevant to the discussion of a simplified instructional agent that helps trainees learn how to classify aircraft on the basis of the type of radar they employ.

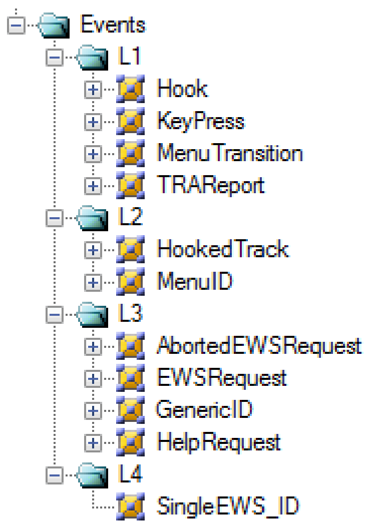


Figure 10-3. An event abstraction hierarchy displayed in a component of the RML authoring environment

The event abstraction hierarchy in Figure 10-3 consists of four levels:

L1: Events inserted into the event cloud by the adapter translating state changes in the AAWC task application into recognizable events. Events produced by the AAWC task application indicating aircraft selection events (*Hook*), key press events (*KeyPress*), control menu transition events (*MenuTransition*), and aircraft information display events (*TRAReport*) are translated by the AAWC task adapter and inserted into the event cloud.

L2: Events inserted into the event cloud by behavior models monitoring simple trainee actions. Behavior models translate fleeting *Hook* and *MenuTransition* events into persistent *HookedTrack* and *MenuID* events to maintain information about currently selected aircraft and the current task menu state in the event cloud.

L3: Events inserted into the event cloud by behavior models monitoring unit task-level trainee actions. Each event represents a sequence of L2 *HookedTrack* and *MenuID* events produced during fundamental information gathering and aircraft classification part-tasks required of the AAWC trainee. A behavior model monitoring *HookedTrack* and *MenuID* events traces trainee "electronic warfare signal" (EWS) activities and inserts *EWSRequest* and *AbortedEWSRequest* events into L3. A separate behavior model monitoring trainee classification activities inserts *GenericID* events into L3.

L4: The *SingleEWS_ID* event is inserted into the event cloud by a behavior model monitoring strategically sequenced EWS and aircraft classification unit tasks.

The event abstraction hierarchy allows the instructional agent to comprehend activities occurring in the AAWC task at four levels of abstraction. Events in each level of the abstraction hierarchy are derived from behavior models that aggregate, correlate, or abstract events in the previous level of abstraction. For example, information about an aircraft's radar informs trainee decisions about its intent and type. To obtain information about an aircraft's radar, a trainee must select it and precipitate a series of menu transitions (through key presses). L1 events resulting from these actions will, in turn, lead to L2 *HookedTrack* and *MenuID* events. These L2 *HookedTrack* and *MenuID* events can be correlated and aggregated in a behavior model capable of asserting L3 *EWSRequest* and *AbortedEWSRequest* events into the event cloud. To illustrate how the abstraction of events can be increased in the event cloud, a behavior model capable of increasing event abstraction from L2 to L3 are now discussed.

Tracing Trainee Actions with Behavior Models

RML behavior models are automata whose nodes are labeled states indicating the steps of the represented behavior and whose edges are event patterns or event pattern rules. Edges based on event pattern rules allow a behavior model to transition states and insert new events into the event cloud. Figure 10-4 shows a behavior model that enable the AAWC instructional agent to monitor trainee EWS request activities. The figure includes (1) a single *start* state; (2) a single *Select EWS Request* state; (3) a single *stop* state; (4) a single *track* variable used to hold information about the aircraft hooked prior to the initiation of the EWS request; and (5) four named edges, each embellished with a circular check mark.

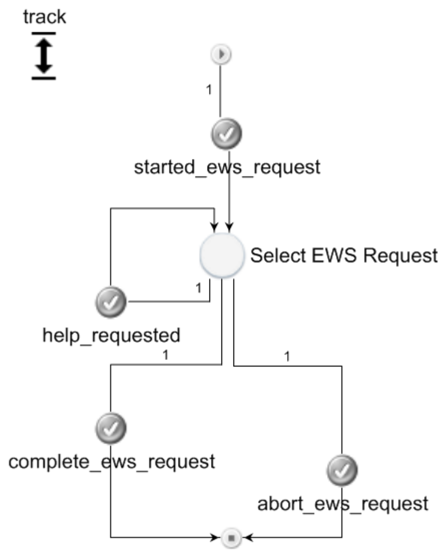


Figure 10-4. RML behavior model capable of monitoring trainee EWS request activities

The formal attributes of event patterns and event pattern rules are specified graphically in the GME-based authoring environment. The event pattern constituting the *start_ews_request* edge of the behavior model in Figure 10-4 is shown in Figure 10-5. The event pattern consists of the following elements: (1) a single conjunction indicator; (2) three event templates labeled *menu*, *hooked*, and *key*; (3) a single integer literal *0*; (4) a single string literal *"f10"*; and a reference to the *track* variable. The conjunction indicator is connected to all three of the event templates indicating that they must all be co-present in the event cloud for the pattern to match. Events are represented as rectangular "containers." Event attributes are visible as labeled "ports" in the containers. The *0* literal is connected to *menu.id* via a connector terminating with an open diamond. This type of connector is used to indicate correlations between event properties and variables, literals, or other event properties. Note that attribute correlation connections are based on comparison operators (=, !=, <, >, <=, >=, etc.). The *"f10"* literal is correlated to *key.key*. The *hooked.tra* event property is connected to the track variable reference via a connector terminating with a solid circle.

This type of connector indicates an assignment. In this instance, the connector indicates that the track variable is to be assigned the value of the *hooked.tra* event property. Essentially, the event pattern shown in Figure 10-5 specifies a template that will enable an instructional agent to notice when the trainee is in *menu.id = 0* (the top menu), an *aircraft is hooked*, and the *"f10" key is pressed*. This event conjunction will enable the behavior model shown in Figure 10-4 to transition to the *Select EWS Request* state and bind the hooked aircraft to the locally scoped *track* variable.

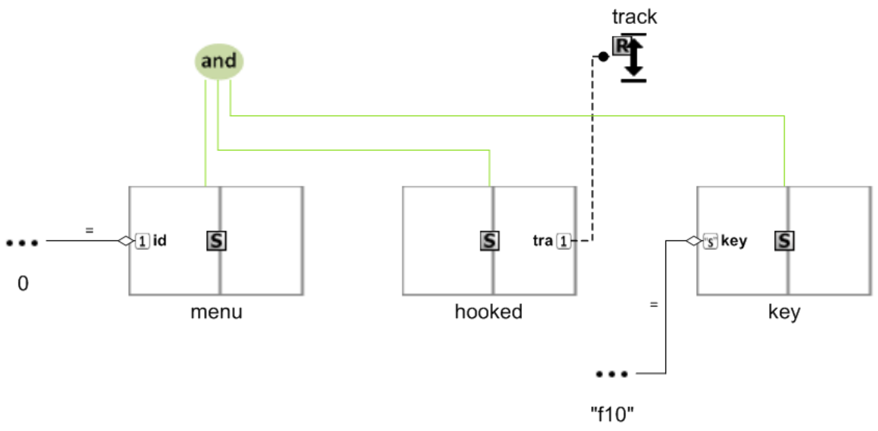


Figure 10-5. RML *start_ews_request* event pattern

The *complete_ews_request* edge in Figure 10-4 is shown in Figure 10-6. The event pattern rule essentially requires that an *"f1" key* be pressed when an aircraft with an id equal to *track* is *hooked*. Notice how the value of *track* is correlated to *hooked.tra* and how *hooked.tra* is used to assign a value to the *tra* property of a new *EWSRequest* event. Note that since the pattern rule in Figure 10-6 inserts the new *EWSRequest* into the event cloud, it can be considered functionally equivalent to a production in a rule-based system.

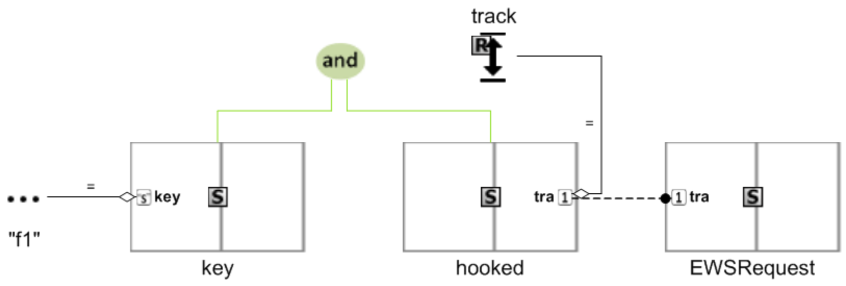


Figure 10-6. RML *complete_ews_request* event pattern rule

RML behavior models can be based on event patterns and event pattern rules capable of matching events from any level of the event abstraction hierarchy. This capability allows a modeler to specify and exploit behavior models that monitor events and trainee activities at high levels of abstraction. The behavior model shown in Figure 10-7 allows the instructional agent to monitor sequences of high-level events indicating that a trainee is efficiently sequencing EWS and aircraft identification tasks.

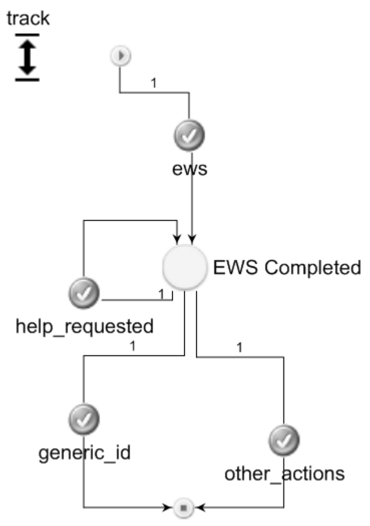


Figure 10-7. RML behavior model capable of monitoring trainee EWS and subsequent identification actions

The "*ews*" edge in Figure 10-7 allows the behavior model to transition from the *start* state to the *EWS Completed* state. The *generic_id* edge allows the high-level behavior model to recognize a completed identification action; insert and even more abstract event into the event cloud, and transition to the *stop* state. Note how the *help_request* edge can detect help request events and deliver precise instruction.

More abstract events inserted into the event cloud indicate that a trainee has successfully performed a sequence of actions indicating they have acquired an AAWC task-critical composite skill. Sequences of events, event patterns, and event pattern rules producing abstract events *operationally define* the abstract events. When abstract events reflect the completion of task-critical actions, the operational definitions underlying the events describe observations or *measurements of performance*. For example, *EWSRequest* event instances added to the event cloud by the event pattern rule in Figure 10-6 can be considered reflections of an operational definition of EWS information request performance based on one aircraft selection action followed by *"f10"* and *"f1"* key press actions. These three events related through patterns and rules in a behavior model, operationally define successful trainee EWS request performances. Events higher in the event abstraction hierarchy provide even more useful information to an intelligent agent. For example, *SingleEWS_ID* events inserted into the event cloud by the behavior model shown in Figure 10-7 indicate that the monitored trainee: (1) has mastered the EWS request process; (2) can remember information about reported radar systems; (3) can correctly classify aircraft on the basis of this remembered information; and (4) has mastered part of the AAWC aircraft identification process.

Transition paths through behavior models leading to new events in the event cloud are formally specified operational definitions of performance. Since behavior models can match against and produce new events at all levels of an event abstraction hierarchy, operational definitions of trainee activity are available at all levels of task hierarchy. Instructional agents developed in RML capitalize on this by monitoring the frequency and recency of events using an extension of the General Performance Equation (GPE) (Anderson & Schunn, 2000; Jastrzembski, et al., 2006). Instructional agents can be configured to monitor task or trainee generated events and use extensions of the GPE to assess and predict performance.

Using CDOs to Evaluate the Appropriateness of Trainee Actions

RML intelligent agents can effectively use behavior models to trace trainee actions. Behavior models are particularly effective in task contexts where it is relatively easy to capture anticipated (correct or incorrect) sequences of action. In training contexts where multiple actions are appropriate, it can be difficult to specify RML behavior models covering large spaces of alternative trainee actions. Intelligent agents executing in the CECEP architecture use CDOs to assess the appropriateness of trainee actions under these circumstances. This section demonstrates how a CDO can be used to capture structural and relational domain knowledge in such a way that constraint-satisfaction processes in soaCDO allow an RML instructional agent to (1) make sense of a trainee's actions and intentions; and (2) determine the appropriateness of a trainee's actions and intentions. Capturing and processing domain knowledge this way in the CECEP architecture greatly reduces the burden of behavior model specification.

Figure 10-8 shows a CDO capturing structural domain knowledge related to the AAWC task described in the previous section. CDOs capture *structural domain knowledge* in tree-like structures consisting of the following:

- *Entities*: Indicated by gray circles. Entities represent domain constructs. In Figure 10-8, entities describe the structural attributes of the set of *track* (or aircraft) entities that a trainee is tasked with classifying. Note that in CDOs, the root entity determines the entity set of interest.

- *Structural Decompositions*: Indicated by rectangles labeled "and." Decompositions represent fixed entity sub-structures. In Figure 10-8, a *track_decomposition* indicates that *track* entities are comprised of *position*, *movement*, *ews*, *model*, and *assessment* sub-entities. Additionally, *assessment* entities are comprised of *threat* and *type* sub-entities.

- *Choices*: Indicated by rectangles labeled "xor." Choices represent alternative entity sub-structures. In Figure 10-8, *ews_choices*, *model_choices*, *threat_choices*, and *type_choices* capture alternative sub-entity choices the trainee will ultimately have to choose between. For example, to classify a track, the trainee will likely have to determine its *ews* choice and certainly have to determine/assume its *threat* and type *choices*.

- *Entity Properties*: Indicated by attached "~" values. Attached values represent entity properties.

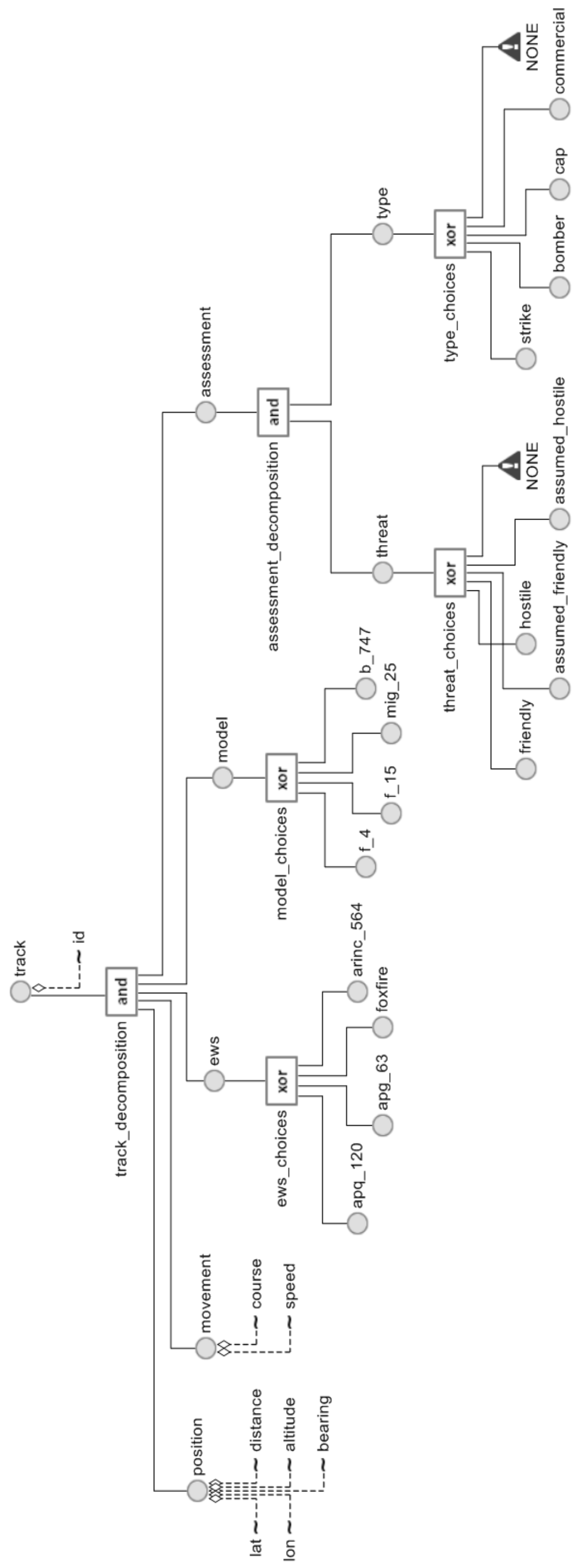


Figure 10-8. A CDO capturing knowledge about *track* entities specified in RML

CDOs additionally capture *relational domain knowledge* in a constraint language. Domain-specific constraints specified in the constraint language express complex relationships between entities represented in the tree-like structure. Tables 10-1 lists example constraints that capture a sub-set of the classification "rules" AAWC task trainees were endeavoring to memorize and act upon.

Table 10-1. Domain-specific constraints integrated into the *track* entities cognitive domain ontology

Name	Specification
C1	`iff ews_choices is arinc_564 then model_choices is b_747`
C2	`iff ews_choices is apq_120 then model_choices is f_4`
C3	`iff ews_choices is apg_63 then model_choices is f_15`
C4	`iff ews_choices is foxfire then model_choices is mig_25`
C5	`if model_choices is f_4`
	`then threat_choices is assumed_hostile hostile assumed_friendly or friendly`
	`type_choices is strike`
C6	`if model_choices is f_15`
	`then threat_choices is assumed_friendly or friendly`
	`threat_choices is not (assumed_hostile or hostile)`
	`type_choices is strike`
C7	`if model_choices is b_747`
	`then threat_choices is friendly`
	`threat_choices is not (assumed_hostile or hostile)`
	`type_choices is commercial`
C8	`if model_choices is mig_25`
	`then threat_choices is assumed_hostile or hostile`
	`threat_choices is not (assumed_friendly or friendly)`
	`type_choices is strike`
C9	`if speed is between 350 and 550,`
	`altitude is between 25000 and 36000`
	`then model_choices is b_747`
	`threat_choices is friendly`
	`type_choices is commercial`

CDOs are transformed into constraint networks by soaCDO. These constraint networks are then searched over by a non-deterministic constraint solver. Without additional domain-specific constraints, the search process yields multiple "solutions" consisting of various combinations of choices specified in the original CDO. CDO solutions are conveyed as sets of events into the CECEP event cloud through an adapter. Choices can be limited through "assertions" that essentially insist that a choice have a certain value. When additional domain-specific constraints are incorporated into a CDO, constraint propagation utilizing assertions can produce solutions that can be exploited by an RML model or agent. For example, when an RML instructional agent is executing in the CECEP architecture, it can use task events and trainee actions to assert the values of choices in a CDO. Table 10-2 shows the impact assertions can have.

Table 10-2. Illustration of how *speed* and *altitude* assertions allow an instructional agent to determine the single correct action a trainee should take. Note: solution is an abstraction of events returned by soaCDO

Assertions	`speed is 500`
	`altitude is 30000`
Solutions	`Track`
	`position {~ altitude = 30000}`
	`movement {~ speed = 500}`
	`ews {ews_choices = arinc_564}`
	`model {model_choices = b_747}`
	`assessment {threat_choices = friendly}, {type_choices = commercial}`

Table 10-2 shows how constraint-based search in soaCDO can determine from observed *speed* and *altitude track* values that a trainee should assess the *track* as *friendly* and *commercial*. Notice how the bidirectional implication (iff) underlying C1 in Table 10-1 allows the instructional agent to infer that the trainee will additionally expect the *ews* choice of the *track* to be *arinc_564*. Under these circumstances, the RML instructional agent can assess the adequacy of a trainee's classification action without requiring a specific behavior model. Procedural knowledge that could be captured in a behavior model is instead captured in more flexible structural and relational knowledge in a CDO.

In instructional contexts where ambiguity or equally effective actions impact trainees, it would be virtually impossible to develop an effective RML instructional agent using only behavior models. In such contexts, it is often the case that multiple trainee actions should be considered adequate. Specifying all possible event patterns and event pattern rules under these circumstances would be costly and error prone.

Domain knowledge in CDOs significantly decreases the burden of specifying instructional agents under these circumstances. Table 10-3 shows how constraint-based search in soaCDO can determine from observed ***ambiguous*** *speed* and *altitude* and *ews_choice track* values that a trainee can appropriately classify a *track* 4 ways. Solution events returned by soaCDO can be exploited by "streamlined" behavior models.

Table 10-3. Illustration of how *speed*, *altitude*, and *ews_choices* assertions allow a tutor agent to determine the courses of action a trainee can undertake when the *threat_choices* aspect of a *track* is ambiguous

Assertions	`speed is 500`
	`altitude is 15000`
	`ews_choices is apq_120`
Solutions	`Track`
	` position {~ altitude = 15000}`
	` movement {~ speed = 500}`
	` ews {ews_choices = apq_120}`
	` model {model_choices = f_4}`
	` assessment {threat_choices = hostile}, {type_choices = strike}`
	`Track`
	` position {~ altitude = 15000}`
	` movement {~ speed = 500}`
	` ews {ews_choices = apq_120}`
	` model {model_choices = f_4}`
	` assessment {threat_choices = friendly}, {type_choices = strike}`
	`Track`
	` position {~ altitude = 15000}`
	` movement {~ speed = 500}`
	` ews {ews_choices = apq_120}`
	` model {model_choices = f_4}`
	` assessment {threat_choices = assumed_friendly}, {type_choices = strike}`
	`Track`
	` position {~ altitude = 15000}`
	` movement {~ speed = 500}`
	` ews {ews_choices = apq_120}`
	` model {model_choices = f_4}`
	` assessment {threat_choices = assumed_hostile}, {type_choices = strike}`

To further sharpen an agent's understanding of task ambiguity, additional domain knowledge could be added to the *track* CDO. For example, additional structural and relational knowledge could be used to inform an instructional agent that in ambiguous situations similar to those underlying Table 10-3, a trainee should assign an "assumed" *threat_choice* and take additional actions as an AAWC to acquire additional information and resolve the ambiguity.

Learner Models in RML

According to Bull (2004), "The learner model is a model of the knowledge, difficulties and misconceptions of the individual. As a student learns the target material, the data in the learner model about their understanding is updated to reflect their current beliefs." In this section, three facets of learner models in RML are defined by relating knowledge representations and processes in the CECEP architecture to components of Bull's definition.

In the CECEP architecture, RML instructional agents represent the knowledge, difficulties, and misconceptions of trainees in behavior models and CDOs. RML instructional agents executing in the CECEP architecture trace trainee actions through event monitoring. Technologies underlying the execution of RML models and agents in the CECEP architecture allow behavior models to both *generate* and *recognize* behavior. When used to recognize trainee actions, behavior models support a type of model tracing. Therefore, the *subset of behavior models* an RML instructional agent currently/actively tracing actions constitutes the first facet of the learner model.

Event monitoring by RML instructional agents can also drive a constraint-based domain knowledge mining process. This capability, based on structural and relational constraint knowledge, enables RML instructional agents to develop and refine hypotheses about trainee behaviors. These agents make sense of trainee actions and intentions by contrasting traced actions to hypotheses (explanatory or predictive) about actions and intentions obtained from the mining of domain knowledge. When used to ascertain trainee intentions and evaluate trainee actions, CDOs support a type of trainee understanding. Therefore, the *subset of a solution space* resulting from the constraint-based exploration of a CDO constitutes the second facet of the learner model.

Events operationally defined at any level of an RML instructional agent's event abstraction hierarchy can be treated as performance indicators and analyzed using an extension of the General Performance Equation (Anderson & Schunn, 2000; Jastrzembski, et al., 2006). Events in the CECEP architecture can be produced by behavior models tracing trainee actions or by constraint-based domain knowledge mining processes making sense of trainee actions. Regardless of their source or level of abstraction, all events in the event cloud of an executing RML instructional agent can be analyzed by extensions of the GPE in order to track and predict performance. Therefore, the *set of events being treated as performance indicators* and analyzed using extensions of the GPE constitute the third facet of the learner model.

Summary and Conclusions

The LSCM initiative has invested in the development of authoring environments and advanced cognitive modeling and simulation architectures. These authoring environments allow users to specify models and agents at high levels of abstraction, often using domain constructs directly. The LSCM initiative has developed a DSL called the RML. The RML is tailored to the needs of cognitive scientists developing models and agents that engage in complex event processing. Models and agents specified in RML are transformed by code generators into all code artifacts necessary for execution and simulation in a net-centric, event-driven CECEP architecture.

RML agents represent and process declarative, procedural, and domain knowledge. To allow agents to represent and process declarative knowledge, the CECEP architecture includes an associative memory system functionally equivalent to the declarative memory system in the ACT-R cognitive architecture. To allow agents to represent and process procedural knowledge, the CECEP architecture includes behavior models, a type of automata representing behavior above the level of conventional rule-based systems.

Finally, to allow agents to represent and process domain knowledge, the CECEP architecture includes a constraint-solver.

This chapter described cognitive modeling and simulation in the LSCM initiative. The chapter demonstrated how LSCM instructional agents track and make sense of learner actions using behavior models and CDOs. The chapter finally described how behavior models, CDOs, and events tracked using a performance monitoring model based on the GPE (Anderson & Schunn, 2000; Jastrzembski, et al., 2006) make up the facets of *learner model in* RML.

Two things conspicuously missing from the LSCM/RML cognitive modeling and simulation framework are (1) constraints reflecting rigorous instructional science and (2) the support of a general learning management and instructional delivery system. While RML instructional agents can use behavior models, CDOs, and derivatives of the GPE to build student models that monitor/predict trainee actions, intelligent agent authors are ultimately responsible for the instructional science underlying their agents. Simply put, this means that researchers developing instructional agents using RML must work with subject matter and educational experts to ensure that their agents instruct effectively. Lacking a comprehensive learning management system, the LSCM/RML cognitive modeling and simulation framework is fundamentally dependent on complementary capabilities such as the ARL GIFT. Researchers developing intelligent tutors in RML conceive of CECEP as a technical solution to the problem of monitoring trainee actions and delivering contextually relevant instruction. These same researchers conceive of GIFT as a broader ITS framework that will provide critical services missing in RML and CECEP.

References

Anderson, J. R., Bothell, D., Byrne, M. D., Douglass, S. A., Lebiere, C. & Qin, Y. (2004). An integrated theory of the mind. *Psychological Review 111, (4),* 1036-1060.

Anderson, J.R. & Schunn, C. D. (2000). Implications of the ACT-R learning theory: No magic bullets. In R. Glaser (Ed.), *Advances in instructional psychology: Educational design and cognitive science (Vol. 5),* (pp. 1-34). Mahwah, NJ: Erlbaum.

Bull, S. (2004). Supporting Learning with Open Learner Models. *4th Hellenic Conference with International Participation: Information and Communication Technologies in Education*, Athens.

Douglass, S. A. & Mittal, S. (2013) A Framework for Modeling and Simulation of the Artificial. Andreas Tolk (Ed.), *Ontology, Epistemology, and Teleology of Modeling and Simulation*, Intelligent Systems Series, ISRL 44, (pp. 271-317), Springer-Verlag.

Douglass, S. A. & Myers, C, W. (2010). Concurrent knowledge activation calculation in large declarative memories. In *Proceeding of the 10th International Conference on Cognitive Modeling – ICCM2010,* Philadelphia, PA.

Fu, W.-T., Bothell, D., Douglass, S. A., Haimson, C., Sohn, M.-H. & Anderson, J. A. (2006), Toward a Real-Time Model-Based Training System. *Interacting with Computers, 18(6),* 1216-1230.

Hansen, H. & Douglass, S. A. (submitted). RML/NERML: Semantically anchoring behavioral models in a complex event processing architecture. In *Proceedings of the 12th Workshop on Domain-Specific Modeling,* OOPSLA/SPLASH 2012, Tucson, AZ.

Jastrzembski, T. S., Gluck, K. A. & Gunzelmann, G. (2006) . Knowledge tracing and prediction of future trainee performance. *In Proceedings of the interservice/industry training, simulation, and education conference (IITSEC).* Orlando, FL: National Training Systems Association (pp. 1498 – 1508).

Ledeczi, A., Maroti, M., Bakay, A., Karsai, G., Garrett, J., Thomason, C., Nordstrom, G., et al. (2001). The generic modeling environment. In *Proceedings of the Workshop on Intelligent Signal Processing*, Budapest, Hungary (Vol. 17).

Luckham, D. C. (2002). *The Power of Events: An Introduction to Complex Event Processing in Distributed Enterprise Systems*. Addison-Wesley Longman Publishing Co.

Molnár, Z., Balasubramanian, D. & Lédeczi, A. (2007). An Introduction to the Generic Modeling Environment. In *Proceedings of the TOOLS Europe 2007 Workshop on Model-Driven Development Tool Implementers Forum*. Zurich, Switzerland.

SECTION III

EMERGING LEARNER MODELING APPROACHES

X. Hu, Ed.

CHAPTER 11 –Emerging Learner Modeling Concepts

Xiangen Hu and Donald M. Morrison
University of Memphis - Institute for Intelligent Systems

Introduction

While the authors of previous chapters have addressed fundamental issues of learner modeling such as common limitations and challenges (Section I) and state-of-the-art implementations (Section II), the authors of chapters in this section present arguments that raise, in various ways, an important issue that is central to the whole notion of a general-purpose ITS framework—the nature of the relationship between the learner model and domain model, and the possibility of defining these models in a way that allows for transportability across systems, or at least allows for different systems to communicate with each other intelligently about what different learners know and are able to do.

The idea that shareable models might exist, or ought to, emerges from the following observations. First, systems based on the same architecture and underlying cognitive theory have been successfully implemented in different domains. For instance, the dialog-based AutoTutor architecture has been used to support learning in physics, computer science, and research design; the model-tracing approach used in Cognitive Tutor supports different domains within mathematics; and the constraint-based tutors developed at the University of Canterbury, New Zealand, have been used in domains as various as electronics, discrete mathematics, language learning, and Newtonian physics. Second, systems based on different theories and architectures have been developed to support learning in the same domain. For example, Cognitive Tutor and ALEKS are based on different architectures and underlying theories of learning, but both have demonstrated effectiveness in support of mathematics. Third, similar learner models, pedagogical strategies, and forms of knowledge representation are observed in different ITS implementations. For example, all ITSs find a way to measure the learner’s current state of knowledge against some ideal state, ask questions or help the user solve problems designed to help her move in the direction of the ideal state, and keep track of progress over some period of time.

Given these observations, it is natural to consider whether there might be some universal principles of learner modeling that could be generally applicable. The practical advantages of a universal learner model, and the implications for a general-purpose framework such as GIFT, are obvious. For one thing, the ability to import a learner model from one ITS into another would help to solve the “new-user” problem, i.e., how to efficiently initialize the model when an ITS first encounters a learner. For example, an intelligent calculus tutor would obviously benefit from knowing an individual learner’s knowledge of algebra, assuming this knowledge came in a form it could understand. An ITS-independent learner model would allow the ITS to initialize its local model by importing relevant portions of a learner model from another ITS, or perhaps from a central learner model server or learning management system. Also, to the extent that the learner model is in some sense an “overlay” of the domain model, a transportable learner model might imply the existence of a universal structure for domain models. Given the two in combination, it would possible to test the relative efficacy of different pedagogical approaches across different ITSs, thereby advancing our scientific understanding and technical expertise in ways that are not currently possible.

Chapter Summaries

The chapters in this section approach the problem in different ways. Robson, Hu, Morrison, and Cai (*The Need for a Mathematical Model of Intelligent Tutoring*) take a highly theoretical approach, imagining the existence of a unified mathematical model in which the learner’s current cognitive and affective state is represented mathematically as a finite set of state variables, named *S*. A model of this type makes it possible to measure the precise distance between a learner's current state and some ideal state. Further, a unified mathematical model could be used to model not just the learner's domain knowledge, but other factors that influence learner performance such as affect, motivation, and social environment. Within a framework such as GIFT, the learner model could be represented mathematically, with the other components of the systems providing mechanisms for reading, manipulating, and outputting values for the set of state variables.

In their chapter, Rus et al. (*Towards Learner Models based on Learning Progressions in DeepTutor*) discuss the implications of learning progressions (LPs), an approach to domain modeling that has gained currency over the past decade, primarily in the science education community. Unlike curriculum standards devised by panels of domain experts that define what students ought to know and be able to do at certain grade levels, LPs are empirically based descriptions of the paths that learners tend to take from naive to more expert understanding of central concepts in a domain. As such, Rus et al. argue, LPs fit well with the central purposes of an ITS, which seeks to identify a learner's current level of understanding, then help that person move along to the next level. As an example, the authors describe a LP in Newtonian physics, which forms the basis for DeepTutor, the first LP-based ITS. DeepTutor's LP is two-dimensional, where one dimension represents seven strands ("big ideas" such as Kinematics, Force and Motion, Newton's Third Law), and the other up to seven progressive levels of understanding within each strand, based on empirical studies of student learning. The result is a reasonably granular representation of domain knowledge that can be used to identify a student's current state of understanding and select a topic for tutorial discourse that can serve as a bridge to the next level. Importantly, the learner and domain models are tightly linked, with learner model becoming a set of pointers to a LP, which constitutes the domain model.

In their chapter (*Modeling Student Competencies in Video Games Using Stealth Assessment*) Shute, Ventura, Small, and Goldberg consider the use of "stealth assessments" embedded within the structure of a serious game. As an example, the authors describe the case of Newton's Playground, a game that challenges players to guide a green ball to a red balloon using combinations of inclined plane/ramps, pendulums, levers, and springboards. Because all of the objects obey the basic rules of physics relating to gravity and Newton's three laws of motion, success depends on an intuitive understanding of these principles. To the extent that a set of embedded assessments can infer a learner's developing mastery of these principles from the learner's gameplay, the game combines informal learning with assessment of a learner's current state of conceptual understanding. The authors report the results of an experiment in which a sample of 165 middle-school students played the game over a period of six 45-min sessions, totaling approximately 5 hours. Learning gains from pretest to post-test were significant (t (154) = 2.12, $p < 0.05$). Importantly, the authors claim that their "stealth" assessment indicators (based on analysis of the log files generated from user's gameplay) correlate with a player's knowledge of physics concepts as measured by the tests.

In *Knowledge Component Approaches to Learner Modeling*, Aleven & Koedinger describe the construction of learner models organized around collections of interrelated knowledge components (KCs), where a KC is defined as "an acquired unit of cognitive function or structure that can be inferred from performance on a set of related tasks." An example is the ability to compute the area of a circle given the radius. Expertise in a given domain is performance-based, defined not so much as what a person knows, but what the person is able to do; thus, a given domain is defined as the set KCs that are required to solve a finite set of problems (or complete tasks) of particular types. Note that here again we see the close connection between the learner model and the domain model within the context of an ITS. The learner model is the system's estimate of the subset of KCs within the domain the learner has mastered, and the "psychological reality" of the model is its ability to predict the student's performance on given tasks within the domain of interest. Given this framework, the authors discuss what they identify as three major challenges inherent in the KC approach: the challenge of building “psychologically-real” domain models from scratch, determining the learner’s level of mastery of the component KCs, and, given an accurate learner model, determining how best to tailor instruction in a way that helps the learner master those KCs that remain to be learned. The paper focuses primarily on the first challenge, the notoriously labor-intensive process of authoring new domain models. The authors argue that so-called “manual” approaches, in which domain experts identify component KCs using a process of cognitive task analysis, provide useful first-order descriptions. However, automated approaches to creating and refining KC models can substantially improve on models initially developed using “manual” approaches.

Broadly speaking an ITS’s learner model is a formal representation of a learner’s cognitive and affective state. In the final chapter in this section (*Assessing the Disengaged Behaviors of Learners*), Baker and Rossi focus on what is arguably the messier of these two components – the learner’s affective state, particularly as evidenced by off-task behavior, gaming the system, carelessness, and what the authors term “Without Thinking Fastidiously (WTF) behavior. While such behaviors are probably easy enough for humans to observe in real time, the trick is to devise machine-based detectors that are capable of making reasonably accurate assessments of the degree to which learners are engaging in these behaviors at any given point in a tutorial session. The resulting data can be used to modify instructional strategies on the fly, and, perhaps more importantly, to provide useful feedback to system developers on the relative effectiveness of different strategies in maintaining student engagement, and help researchers better understand relationships among variables such as a student’s off-task behaviors, attitudes toward computers in general and automated tutors in particular, attitudes toward the domain, and academic drive. A number of algorithms and approaches that have been found to effective in detecting these forms of disengaged behavior are described. For example, one of the authors (Baker) built an automatic off-task-behavior detector based on multiple inputs such as the difficulty of a given interaction, and how many standard deviations longer or slower it took the student to complete the action compared to other students.

Recommendations for GIFT

A theme running through a majority of the chapters has to do with various possible uses of log data, i.e., records of the tutor-learner interactions, what Robson et al. refer to as “the paths traced through the learner state data.” Many of the authors recommend that their own tools be integrated into GIFT for this purpose. For example, Baker and Rossi suggest that GIFT could provide support for the construction of learner engagement detectors by creating API-level links between GIFT and the educational data mining (EDM) Workbench. As a result, data could be pulled directly from GIFT, and detectors exported back to GIFT for use in the learner model. The authors also recommend that GIFT incorporate tools for labeling engagement data, especially through the use of “text replays,” i.e., re-runs of interaction logs that allow for the careful study of these interactions for the purpose of system improvement. This feature is also provided by the Workbench. Similarly, Aleven and Koedinger recommend that, as a first step, participating ITSs log student-tutor interactions in the format employed by DataShop, the data mining tool used with the Cognitive Tutor. For their part, Shute and her colleagues recommend the future developmental efforts of GIFT should aim at identifying authoring tools and methods that ease the process of embedding stealth assessment capabilities in whatever game-based learning environments are being employed, because different games link game performance with demonstration of targeted expertise in different ways. To solve this problem, Shute et al. recommend that GIFT incorporate a “Gateway Module,” which would be responsible for rapidly translating performance in a given game to a data structure that could be understood from within GIFT’s domain model. Rus and company take a somewhat different approach, focusing on modifications that would be necessary if their own ITS, DeepTutor, were to be in some way integrated into the GIFT framework. Although DeepTutor also employs a modular design, the architecture is fundamentally different; for example, whereas in GIFT the domain module performs assessments, this function is performed by a separate module in DeepTutor. Also, the

DeepTutor learner model does store information about levels of domain understanding separately, but instead maintains a set of pointers to the learning progressions in the domain model; in other words, the learner and domain models are more tightly coupled than is currently specified in GIFT. To get around this problem, the authors suggest that GIFT provide different specifications for different types of ITS.

Read together, the chapters in this section highlight the importance of the effort to better understand the relationship between learner and domain models, and find ways of sharing models across systems. For example, given that both DeepTutor and Newton’s Playground share a common domain, the principles of Newtonian physics, it is natural to consider the possibility of using the two in combination. It would be interesting to see, for example, whether learners who receive tutoring from DeepTutor and then go on to play the game are more successful than those who have not been tutored, or, conversely whether DeepTutor finds that students who have played the game are farther along on a relevant learning progression than a similar group of students who have not played the game. However, it is even more interesting to consider how the relationship between the two might work in a world of shareable domain and learner models. For example, given a common learner model with pointers to a domain model organized as a set of learning progressions in Newtonian physics, Newton’s Playground might be able to give a learner problems consistent with her current level of knowledge, while DeepTutor could offer a tutorial dialogue designed to help a learner work through misconceptions that might have been revealed by the stealth assessments embedded in the game. In this scenario, a framework like GIFT might support the collaboration between the game and the tutor primarily by providing a framework for communication of learner data between the two systems. In other words, the domain model would not need to be incorporated into GIFT itself so long as DeepTutor and Newton’s Playground agree to maintain learner models that point to the same domain model, perhaps housed on a separate, web-based domain model server.

CHAPTER 12 –The Need for a Mathematical Model of Intelligent Tutoring

Robby Robson[1], Xiangen Hu[2], Donald M. Morrison[2], Zhiqiang Cai[2]
[1]Eduworks Corporation; [2]University of Memphis

Introduction

ITSs are a large family of technologies that attempt to replicate the experience and learning gains derived from one-on-one human tutoring (for recent reviews, see Durlach & Ray, 2011; Graesser, Conley & Olney, 2012). While these systems are notoriously diverse in their functionality and construction, several authors have created abstract models of ITSs that attempt to capture common components, generally accepted to be the learner model (sometimes called the "student model"), the domain model, the tutor model, and the user interface (Durlach & Ray, 2011, Elson-Cook, 1993; Graesser et al., 2012; Nkambou, Mizoguchi & Bourdeau, 2010; Psotka, Massey & Mutter, 1988; Sleeman & Brown 1982; VanLehn, 2006; Woolf, 2008). ITS have also been classified into categories such as model-tracing, example-tracing, constraint-based, and dialogue-based tutors. Finally, VanLehn has observed that in describing an ITS, it is useful to distinguish between an outer loop of a tutor, which focuses on task selection and macro-adaptation, and the inner loop which handles in-task interactions and micro-adaptation (VanLehn, 2006).

Gaps

These descriptions, however, are largely *qualitative* descriptions that help researchers understand how a given tutor works, but do not help answer *quantitative* questions about how well they work and ultimately how they can be improved. In this chapter, we consider what type of model is needed for answering quantitative questions such as the following:

- What is the gap between the behaviors exhibited by a student and by an expert?
- At what rate is an ITS closing the gap?
- How accurately does an ITS assess knowledge, skills, affective states, and other attributes of a learner?
- Among all possible strategies that an ITS has available, which are likely to lead to the most learning gains in a given situation?
- Given two strategies, which leads to desired learning outcomes in the least amount of time?
- Among the algorithms used by the inner and outer loops of a tutor, where will improvements have the greatest impact?

Most of these questions require measuring two basic quantities about an ITS: Its effectiveness in helping learners achieve learning goals and how efficiently it helps students achieve them. To date, effectiveness is usually measured in terms of the effect size (Cohen's *d*) of learning gains. While aggregate, summative measures such as these are useful in determining how well a tutor is working, they do not tell us what caused the learning, what is happening as learners use the tutor, or how we can improve the tutor. Efficiency, if it is measured at all, is measured by comparing how long it takes to learn concepts or behaviors using an ITS to how long it takes to learn them using other instructional means. Once again, this does not help us make improvements because we don't know what in the tutor causes faster or slower learning, and it does not give us any information about an ideal efficiency against which we can measure how well a tutor is working. To answer these questions, we need models that enable us to measure learner progress and learner rate of progress towards a goal as the tutor operates. The word "measure" is key here and implies quantification.

Exploration of the General Model

Two Analogies

Before discussing the quantification problem, we present two analogies that may help explain why a formal mathematical model is so important. The first is mechanical and views the problem of tutoring a student as analogous to using an autopilot to guide an airplane. As with an ITS, an autopilot is a machine substitute for a human. Its goal is to steer an aircraft to a destination, much as the goal of an ITS is to guide a learner to a learning objective. The autopilot operates using real-time data on position, atmospheric conditions, trim of the aircraft, and other factors and attempts to follow a flight path, which, in many cases, has been calculated to maximize efficiency given constraints such as atmospheric conditions, terrain, the flight paths of other aircraft, the cost of flying at different altitudes, and the ultimate destination. It controls the path by adjusting the ailerons, elevators, rudder, and thrust.

We can think of the data that defines the current state of an aircraft as analogous to the learner model that defines the current state of the learner. But whereas an autopilot continually measures this state and follows a path, ITSs typically measure only the initial and end states, or possibly a small number of intermediate states. This is not sufficient to quantify the dynamics of an ITS or create the analog of a guidance system, which is what we want.

Of course, a learner is not a machine, and an autopilot does not teach a plane to fly. A better analogy, in some ways, is that of a doctor treating a patient. Unlike the autopilot, the doctor, even with modern imaging technologies, cannot observe the precise state of a system as complex as the human body. At best the doctor can talk to the patient, take the patient's history (or look it up), conduct a physical examination, if necessary order some lab tests, and then, using knowledge of the relationship between clusters of symptoms and disease processes, make some guess about what is likely going on. If the doctor finds that the patient is unhealthy, the goal is to find a course of treatment, which, if followed, will return the patient to a state of health, but since the doctor cannot know the actual state of the patient, in practice, the doctor can, at best, monitor the patient's symptoms, with the expectation they will go away. If this is achieved, the patient is presumed to be healthy.

Although an ITS operates more like an autopilot in the sense that it uses a guidance system to steer learners towards a goal, it operates more like a doctor in that it cannot measure the precise state of learner and must rely on models of generic learners to interpret the measurements made of any specific individual. The ITS must therefore rely on a set of measurements and models of how humans learn to infer the state of the learner and prescribe interventions that change the state to a desired one, usually the state of an "expert." However, to discover which interventions should be prescribed, the ITS must have a means to observe how (and how much) the inferred learner state changes in response to specific interventions. This requires a well-defined mapping between what an ITS can measure and a model of the learner state. In today's practice, it is often difficult to specify what measurements an ITS is taking and how those are being translated into a model whose distance from an ideal model can be measured.

A Conceptual Model for Tutoring

Although ITS developers employ different approaches to the problem of guiding human learning, there is a great deal of similarity in how their systems function. Conceptually, each ITS uses interactions between the student and the system (verbal, written, haptic, biometric, etc.) to infer and alter what the learner knows or can do. The ITS observes these interactions and translates them into machine-readable data. These data, and possibly other data that are known to the ITS, are used to estimate the learner's current state of knowledge and skill. Knowledge states are typically represented as mastery levels of concepts or skills, knowledge components (Koedinger, Corbett & Perfetti, 2012), or a similar set of parameters. In addition to using estimates derived from interaction data, some ITSs use the structure of the knowledge domain and history of learner's interactions to make inferences about knowledge states, e.g., by inferring that if a learner has demonstrated mastery of concept C or successfully performed task T several times, then the learner has also mastered concept D and can perform task U. In some instances, affective states are estimated by the ITS in analogous ways. Once the state of the learner is estimated, the ITS uses this state to determine what interaction or interactions will next take place.

It is useful to separate the foregoing description of an ITS into two parts:

1. *State modeling and estimation*: Through the use of data collection devices and strategies such as emotion sensors, observation of game performance, responses to test questions, and direct questions (in the case of dialog-based systems), the system estimates where the learner currently lies in a multidimensional model of possible cognitive and affective states.

2. *Evaluation and decision making*: Given the model of possible states and the estimate of the learner's current state, the system decides that either (1) the learner's state is optimal with respect to normative expectations, in which case it moves to the next step in the outer loop; or (2) the learner's state is suboptimal, in which case it selects and enacts what it considers to be the best intervention in the inner loop.

State modeling and estimation is the general form of learner modeling, and the second step is the functional view of how an ITS uses expert, domain, and pedagogical models to direct its operation based on a state model. The cycle of estimation, evaluation, and decision making is repeated until the outer loop is exited.

Ultimately, we want to evaluate how well the tutor is working and how to improve it by improving its ability to estimate, evaluate, and change the learner's state. For this purpose, a slightly more formal formulation is needed and can be given as follows:

1. A learner model (the learner's present state) can be represented by a (finite) set of state variables. Each ITS represents a learner's state by the values of these variables as they vary over time. In more formal terms, there is a space *S* that represents all possible states of a learner. These states can ostensibly include motivational, affective, cognitive, and social factors ranging from mastery levels of domain concepts to frustration and motivation levels and certain individual or cultural beliefs, which might affect how a learner approaches a task (e.g., see Arroyo et al., 2009; D'Mello & Graesser, 2010). In the autopilot analogy, *S* is the position of the aircraft and in the patient analogy, *S* is the actual state of the patient (which can only be inferred and not directly measured).

2. The state of a learner at any time *t* is estimated based on a set *O* of observable variables. These are obtained through interactions with the ITS or through data communicated to the ITS from other systems, e.g., from a LMS or a game operating in a multi-system framework such as GIFT

(Sottilare, 2012). The ITS contains an algorithm *a* that maps the history of observations about a learner to the learner's current state. This is a function $a{:}O \times T \rightarrow S$, where *T* is the time interval during which observations are available. The variables in *O* can be thought of as the symptoms and measurements taken by the doctor, and *a* is the process the doctor uses to infer the state of the patient. (Note: The function *a* may, in practice, use past values of *a* in computing the present value of *a*. In other words, the history of estimates of the learner's state may be used to estimate the current state.)

3. Within *S* there are target states, and the goal of the ITS is to move the learner's current state to a target state. In the analogies, these are the destination of the aircraft and the state of health of the patient, which is a range of states. In an ITS, these may be defined by an expert model or the expected behavior of a learner at a particular developmental stage.

4. The ITS functions by doing the following:

    a. Interacting with the learner.

    b. Measuring the values of variables in *O.*

    c. Applying *a* to estimate the learner's state in *S.*

    d. Selecting a strategy and associated actions that will move the learner to the target state.

    e. Implementing those actions through (and only through) a set of interactions with the learner.

Observations about this Model and Questions Raised

All Tutors Trace a Learner Model: Our first observation is that this model is an abstraction of model-tracing tutors but applies to almost all ITSs. For example, the constraint-based tutors described by (Mitrovic, Mayo, Suraweera & Martin, 2001) analyze student answers against a set of constraints to determine the student state and take appropriate actions. The example-tracing tutors of (Aleven, Mclaren, Sewall & Koedinger, 2009) compare student input to correct and incorrect problem-solving behaviors. These constraints and reference examples define points in *S* that the tutor tries to steer toward or away from. Autotutor Lite (Hu et al., 2009) constructs a "learner's characteristic curve" based on semantic comparisons of student input to text that represents expected answers (Robson & Ray, 2012) calculated over a series of turns in the inner cycle. Even if an ITS does not have an explicit expert model or domain model, it computes some set of state variables and takes actions to move the learner state to a desired state. *Conceptually, every ITS is a"learner model tracing" tutor.* Even if different tutors estimate the learner state in different ways and take different actions to change the state, the existence of a parameterized state space makes it feasible to quantify and compare tutors on the basis of how learners move through this space.

Optimal Paths: Our second observation is that the ideal ITS moves a learner along a path in *S* that minimizes "cost" (e.g., time to mastery or actual cost of running a simulation) and maximizes "benefit" (e.g., how close the learner is to the target state and how long the learner will retain that position.) A good test of whether a particular *S* (i.e., a particular set of parameters used to model the learner state) is viable is whether its properties allow for the optimization of paths between any two states with respect to a suitable utility function that reflects cost and benefits. This raises the question of whether existing tutors have either explicit or implicit state models that are sufficient to do this. In most cases, the answer is likely no, and we see this as a limiting factor to making progress in the area.

We observe that most existing tutors focus on cognitive (and possibly affective) state variables that describe learning goals. These state variables are often represented as levels of competency with respect to a set of objectives, knowledge components, or similar constructs, and are implicitly considered to be observable (with error) through assessment. In reality, the situation is more complex, more akin to the patient analogy than the aircraft analogy.

If we picture *S* as a higher dimensional object, then the typical tutor works in a projection of *S* onto a discrete structure (see Figure 12-1). This results in loss of information and makes it impossible to measure the distance between two learner states because there are many points in *S* that correspond to each point in the discrete structure. Moreover, the observability assumption means that, as far as the ITS is concerned learners never stop between observable states. For example, an ITS might observe through assessment that a learner knows or does not know a fact but cannot observe where the learner is in the process of learning the fact. No intermediate state between "not knowing" and "knowing" exists. Even if the discrete structure could be used to estimate distances between states in *S*, this further loss of information makes it hard to determine the learner's path through *S*. ITS authors can program the inner and outer loops of a tutor, and they can empirically determine the effects of this programming on learning outcomes, but they cannot measure the effects on a utility function and cannot determine whether the tutor is close to ideal.

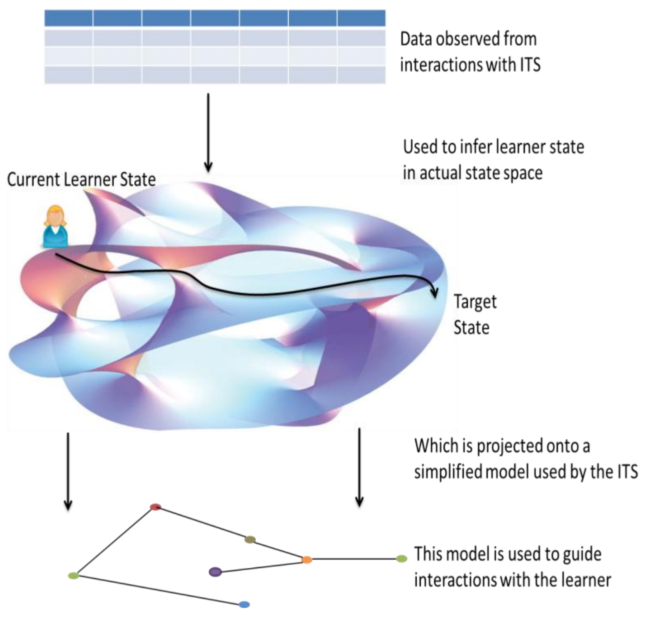

Data observed from interactions with ITS
Used to infer learner state in actual state space
Current Learner State
Target State
Which is projected onto a simplified model used by the ITS
This model is used to guide interactions with the learner

Figure 12-1. How an ITS works*

Universal Components: An obvious question, whose answer has many implications, is whether there is a universal state model. In other words, is there a set of state variables that can be used across multiple tutors? Even if a tutor has a different state space, could it be reduced to a common space without suffering a significant loss in effectiveness? Or, phrased differently, is there universal learner model that can and should be programmed into a framework such as GIFT? As argued in Robson & Barr (Chapter 2 in this volume), this is a key question, and it has been asked in various forms by Durlach (2012), Goldberg, Holden, Brawner & Sottilare (2011), and others.

It is doubtful that a universal state space exists if one includes the cognitive dimension since concepts and knowledge constructs differ from domain to domain. Even should an "ontology of everything" be achievable, it may not be practical to maintain this for all systems at all times. However, if we separate these parameters into a domain model (as is done in GIFT), then it is reasonable to ask whether there is a formulation of domain models that can be used as a template for any *S* and, more importantly, whether motivational, affective and social parameters are sufficiently domain-independent to allow for a manageable set of associated state variables that can be effectively used across most ITSs.

We do not know the answer to these questions, but we observe that affective, motivational, and social components are increasingly being inferred from sensor data and in-system responses (Arroyo et al.,

[*] Surface image from http://en.wikipedia.org/wiki/File:Calabi_yau.jpg.

2009; Calvo & D'Mello, 2010; D'Mello & Graesser, 2010; Robison, McQuiggan & Lester, 2009; Woolf et al., 2009). Since target states in *S* are usually defined in the cognitive or behavioral domain, these components are used primarily as control variables. A universal representation of these components could significantly improve the ability of the ITS to use them to effectively guide learning along optimal paths.

Estimation: The algorithm *a* maps ITS observations to learner states. In real-world examples, *a* operates on data ranging from simulation and game data (Engineering & Computer Simulations, 2013) to data generated by LSA (Wiemer-Hastings, Graesser & Harter, 1998) and by using production rules to analyze student answers (Blessing, Gilbert, Ourada & Ritter, 2009). The challenge of transforming observables into a state model may be equal to (or greater than) the challenge of determining the state model and is just as critical. The ability of an ITS to follow an optimal path is limited by its ability to detect learner state.

Intuitively, an ITS that continually tracks observable state changes along numerous dimensions, including multiple micro-adaptations, has a better chance at accurately estimating the learner's state than an ITS which relies on static models, discrete measurements, and only on macro-adaptations. However, a relatively small number of observables may be sufficient to account for most of the variance in learner state. This has been observed in affect detection (Graesser, Rus, D'Mello & Jackson, 2008) and is a fertile area of research that can be supported by GIFT.

Intervention Selection: Our final observation is that if we understood how different interactions affected the trajectory of the learner in *S*, it would seem relatively straightforward to design algorithms for selecting the best interactions. In other words, assuming we can estimate and evaluate reasonably, empirical experimentation can be used to come up with a potential set of interventions for which selection should be straightforward. Frameworks such as GIFT that can be used to integrate disparate types of interventions are ideal for developing this type of understanding, assuming we a reasonable model of *S* and estimation function *a*.

Future Research and Recommendations for GIFT

To be of general significance, the model presented in this chapter must be more precisely formulated and tested for its ability to model and provide useful insight into existing ITSs. This requires understanding what motivational, affective, cognitive, and social environment data is likely to be represented in *S*, which is perhaps the central problem addressed in this volume. As stated in (Goldberg et al., 2011:10), "*While we intuitively know that it is better to have more information when we are making decisions to tailor instructional feedback and content to individual trainee needs, the influence of specific trainee attributes on instructional decisions can be debated. Additional experimentation is needed to quantify the impact of trainee attributes.*"

This chapter suggests that as this central question is addressed, it will be important to observe the paths traced through the learner state data collected by GIFT and not just the data themselves, and that it will be important to test whether the parameters in a learner model can be used to answer optimization questions. In addition, this chapter points out that GIFT can be used to empirically evaluate the effects of individual interventions, probably at a more granular level than the typical complete ITS, and that a lot of thought should be given to observables. Experiments that investigate what data is required to sufficiently determine affective states provide good models for analyzing data coming out of simulations, games, and sensors with regard to their ability to determine the parameters in *S*.

References

Aleven, V., Mclaren, B. M., Sewall, J. & Koedinger, K. R. (2009). A new paradigm for intelligent tutoring systems: Example-tracing tutors. *International Journal of Artificial Intelligence in Education, 19*(2), 105-154.

Anderson, J.R. (1988). The expert model. In Polson, M. C. & Richardson, J. J. (Eds.). *Foundations of intelligent tutoring systems*. Lawrence Erlbaum. 21-53.

Blessing, S. B., Gilbert, S. B., Ourada, S. & Ritter, S. (2009). Authoring model-tracing cognitive tutors. *International Journal of Artificial Intelligence in Education, 19*(2), 189-210.

Durlach, P. J. (2012). Vanilla, Chocolate, or Chunky Monkey: Flavors of Adaptation in Instructional Technology. *iFest 2012*, from http://www.adlnet.gov/wp-content/uploads/2012/08/Durlach_Adaption_in_IT_iFest-2012.pdf

Durlach, P. J. & Ray, J. M. (2011). Designing adaptive instructional environments: Insights from empirical evidence *Army Research Institute Report*. Arlington, VA.

Engineering & Computer Simulations. (2013). Student Information Models for Integrated Learning Environment (SIMILE) Retrieved February, 2013, from http://www.ecsorl.com/solutions/simile

Goldberg, B. S., Holden, H. K., Brawner, K. W. & Sottilare, R. A. (2011). *Enhancing Performance through Pedagogy and Feedback: Domain Considerations for ITSs*. Paper presented at the Interservice Interindustry Simulation Education and Training, Orlando, FL. https://litelab.arl.army.mil/system/files/IITSEC2011_Goldberg_etal_Enhancing%20Performance%20Through%20Pedagogy%20and%20Feedback.pdf

Graesser, A. C., Rus, V., D'Mello, S. K. & Jackson, G. T. (2008). AutoTutor: Learning through natural language dialogue that adapts to the cognitive and affective states of the learner. In D. H. Robinson & G. Schraw (Eds.), *Recent innovations in educational technology that facilitate student learning* . pp. 95-125. Information Age Publishing.

Hu, X., Cai, Z., Han, L., Craig, S. D., Wang, T. & Graesser, A. C. (2009). *AutoTutor Lite*.

Mitrovic, A., Mayo, M., Suraweera, P. & Martin, B. (2001). Constraint-based tutors: a success story. *Engineering of Intelligent Systems*, 931-940.

Robson, R. & Ray, F. (2012). *Applying Semantic Analysis to Training, Education, and Immersive Learning.* Paper presented at the Interservice/Industry Training, Simulation, and Education Conference, Orlando, FL.

Sottilare, R.A., Brawner, K.W., Goldberg, B.S. & Holden, H.K. (2012). *The Generalized Intelligent Framework for Tutoring (GIFT)*. Orlando, FL: U.S. Army Research Laboratory – Human Research & Engineering Directorate (ARL-HRED).

VanLehn, K. (2006). The behavior of tutoring systems. *International journal of artificial intelligence in education, 16*(3), 227-265.

Woolf, B. P. (2009). *Building intelligent interactive tutors: Student-centered strategies for revolutionizing e-learning*. Burlington, MA: Morgan Kaufmann.

CHAPTER 13 –Modeling Student Competencies in Video Games Using Stealth Assessment

Valerie Shute[1], Matthew Ventura[1], Matthew Small[1], and Benjamin Goldberg[2]
[1]Florida State University, [2]U.S. Army Research Laboratory

Introduction

We have been examining ways to leverage video games to assess and support important student competencies, especially those that are not optimally measured by traditional assessment formats. The term "stealth assessment" refers to the process of embedding assessments directly and invisibly into the learning or gaming environment. Though this approach produces ample real-time data on a player's interactions within the game environment and preserves player engagement, a primary challenge for using stealth assessment in games is taking this stream of data and making inferences about players' competencies that can be examined at various points in time (to see growth) and also at various grain sizes (for diagnostic purposes). In this chapter, we present recent work related to creating and embedding various stealth assessments into *Newton's Playground*, a computer game that emphasizes nonlinear gameplay and puzzle-solving in a two-dimensional (2-D) physics simulation environment. We conclude with a discussion on stealth assessment within GIFT, highlighting research recommendations for enhancing the architecture to support robust methods of evidence-centered assessment.

In this chapter, we examine the possibility of using well-designed games as vehicles to assess and support learning. There are several factors motivating this research. First, our schools have remained virtually unchanged for many decades while our world is changing rapidly. This lack of reform in our schools could be a contributing factor in high dropout rates, especially among Hispanic, Black, and Native American students, which were described as "The Silent Epidemic" in a recent research report for the Bill and Melinda Gates Foundation (Bridgeland, DiIulio & Morison, 2006). According to this report, nearly one third of all public high school students drop out and the rates are higher across minority students. Importantly, when 467 high school dropouts were asked why they left school, 47% of them simply responded, "*The classes were not interesting*." In light of this finding, we need to identify ways (e.g., video games) to get young people engaged in learning the skills needed to succeed in today's competitive economy.

A second reason for using games as assessments is a pressing need for dynamic and ongoing measures of learning processes and outcomes. Interest in alternative forms of assessment is driven by dissatisfaction with and limitations of multiple-choice items. In the 1990s, interest in alternative forms of assessment increased with the popularization of what became known as authentic assessment. Authentic assessment refers to tasks that resemble academic and real-world activities (e,g., Hiebert, Valencia & Afflerbach, 1994). A number of researchers found that multiple-choice and other fixed-response formats substantially narrowed school curricula by emphasizing basic content knowledge and skills within subjects and not assessing higher-order thinking skills (e.g., Kellaghan & Madaus, 1991; Shepard, 1991). However, as Madaus and O'Dwyer (1999) argued, incorporating performance assessments into testing programs is difficult because they are less efficient, more difficult and disruptive to administer, and more time-consuming than multiple-choice testing programs. Consequently, multiple-choice has remained the dominant format in most K–12 assessments in our country. New performance assessments are needed that are valid and reliable, and can be scored automatically.

A third reason for using games as assessment vehicles is that many games typically require a player to apply various competencies (e.g., creativity, critical thinking, problem solving, persistence, and collaboration) to succeed in the game. The competencies required to succeed in many games also happen to be the same ones that companies are looking for in today's highly competitive economy (Arum & Roska, 2011; Gee, Hull & Lankshear, 1996). Moreover, games are a significant and ubiquitous part of young people's lives. For instance, the Pew Internet and American Life Project surveyed 1,102 youth between the ages of 12 and 17. They reported that 97% of youth – both boys (99%) and girls (94%) – play some type of digital game (Lenhart et al., 2008).

In addition to the arguments for using games as assessment devices, there is growing evidence of games supporting learning (e.g., Tobias & Fletcher, 2011; Wilson et al., 2009). However, we need to understand more precisely how and what kinds of knowledge and skills are being acquired. Understanding the relationships between games and learning is complicated by the fact that we don't want to disrupt players' engagement levels during game play. Consequently, learning in games has historically been assessed indirectly and/or in a post hoc manner (Shute & Ke, 2012; Tobias, Fletcher, Dai & Wind, 2011). What's needed instead is real-time assessment and support of learning based on the dynamic needs of players. We need to be able to experimentally ascertain the degree to which games can support learning, and how they achieve this objective.

A challenge with developing a performance-based measure is crafting appropriate situations or problems to elicit a competency of interest. One way to approach this problem is to use video games to simulate problems for performance-based assessment (Dede, 2005; DiCerbo & Behrens, 2012; Quellmalz, Timms, Buckley, Silberglitt & Brenner, 2012). Digital learning environments can provide meaningful assessment environments by providing students with scenarios that require the application of various competencies. In this chapter, we introduce research that explores a variant of this assessment approach by investigating how performance-based assessments can be used in a video game we created called *Newton's Playground.*

Stealth Assessment

Given the goal of using well-designed games to support learning in school settings and elsewhere, we need to ensure that the assessments are valid, reliable, and also unobtrusive (to keep engagement intact). The output from the assessments, however, should be transparent. That is, players should be aware of how they are doing relative to important competencies at any point in time to motivate learning. One way to meet these requirements is to use "stealth assessment" (Shute, 2011; Shute & Ventura, 2013). Stealth assessment refers to Evidence-Centered Design (ECD)-based assessments that are woven directly and invisibly into the fabric of the gaming environment. During game play, students naturally produce rich sequences of actions while performing complex tasks, drawing on the very skills or competencies that we want to assess (e.g., scientific inquiry skills, creativity). Evidence needed to assess the skills is thus provided by the players' interactions with the game itself (i.e., the processes of play), which can be contrasted with a summative score – the norm in educational environments.

Making use of this stream of gameplay evidence to assess students' knowledge, skills, and understanding (as well as beliefs, feelings, and other states and traits) presents problems for traditional measurement models used in assessment. First, in traditional tests, the answer to each question is seen as an independent data point. In contrast, the individual actions within a sequence of events in a game are often highly dependent on one another. For example, what one does in a particular game at one point in time affects subsequent actions later on. Second, in traditional tests, questions are often designed to measure particular, individual pieces of knowledge or skills. Answering the question correctly is evidence that one may know a certain fact: one question – one fact. But by analyzing a sequence of actions within gameplay (where each response or action provides incremental evidence about the current mastery of a specific fact, concept, or skill), stealth assessments can infer what learners know and do not know at any point in time.

Now, because we typically want to assess a whole cluster of skills and abilities using evidence coming from learners' interactions within a game, methods for analyzing the sequence of behaviors to infer these abilities are not as obvious. As suggested above, evidence-based stealth assessments can help address these problems. The next section reviews the game we created called *Newton's Playground* and the associated development of stealth assessments for monitoring learner knowledge and progression.

Stealth Assessment in *Newton's Playground*

Research into what's called "folk" physics demonstrates that many people hold erroneous views about basic physical principles that govern the motion of objects in the world, a world in which people act and behave quite successfully (Reiner, Proffit & Salthouse, 2005). For example, when asked to draw the water level on a picture of a tilted drinking glass, about 40% of young adults draw lines that are not horizontal (McAfee & Proffitt, 1991). When asked to predict the path that a pendulum takes when the string is cut at various points, a large percentage of people make systematically incorrect judgments (Caramazza, McCloskey & Green, 1981). The prevalence of these systematic errors has led some investigators to propose that incorrect performance on these tasks is due to specific "naive" beliefs, rather than to a general inability to reason about mechanical systems (McCloskey & Kohl, 1983). Recognition of the problem has led to interest in the mechanisms by which physics students make the transition from folk physics to more formal physics understanding (diSessa, 1982) and the possibility of using video games to assist in the learning process (Masson, Bub & Lalonde, 2011; White, 1994).

One way to help remove misconceptions in physics is to illustrate physics principles with physical machines (Hewitt, 2009). In physics, a machine refers to a device that is designed to either change the magnitude or direction of a force. Teaching about simple machines (e.g., lever, pulley, and wedge) is widely used as a method to introduce physics concepts (Hewitt, 2009). Recent research on science education also indicates that learners' hands-on experience with such machines (both virtually and physically) support applicable understanding of important physics concepts (Hake, 1998).

We developed a video game called *Newton's Playground* (NP) to help middle school students experience and understand what we call informal physics. We define informal physics as a nonverbal understanding of how the physical world operates. Informal physics is characterized by an implicit understanding of Newton's three laws, balance, mass, conservation of momentum, kinetic energy, and gravity. NP is a 2-D game that requires the player to guide a green ball to a red balloon. The player can nudge the ball to the left and right (if the surface is flat) but the primary way to move the ball is by drawing/creating simple machines (which are called "agents of force and motion" in the game) on the screen that "come to life" once the object is drawn. Everything obeys the basic rules of physics relating to gravity and Newton's three laws of motion. The 74 problems in NP require the player to draw/create four agents: inclined plane/ramps, pendulums, levers, and springboards. All solutions are drawn with colored lines using the mouse.

A ramp is any line drawn that helps to guide a ball in motion. A ramp is useful when a ball must traverse over a gap or obstacle. A lever rotates around a fixed point, that is, a fulcrum or pivot point. Levers are useful when a player wants to move the ball vertically. A swinging pendulum directs an impulse tangent to its direction of motion. The pendulum is useful when the player wants to exert a horizontal force. A springboard (or diving board) stores elastic potential energy provided by a falling weight. Springboards are useful when the player wants to move the ball vertically. For example, in the "golf problem" (see Figure 13-1), the player must draw a golf club on a pin (i.e., little circle on the cloud) to make it swing down to hit the ball. In the depicted solution, the player also drew a ramp to prevent the ball from falling down a pit.

The speed of (and importantly, the impulse delivered by) the swinging golf club is dependent on the size/mass distribution of the club and the angle from which it was released. The ball will then move at a certain speed, length, and trajectory. If drawn properly, the ball will hit the balloon.

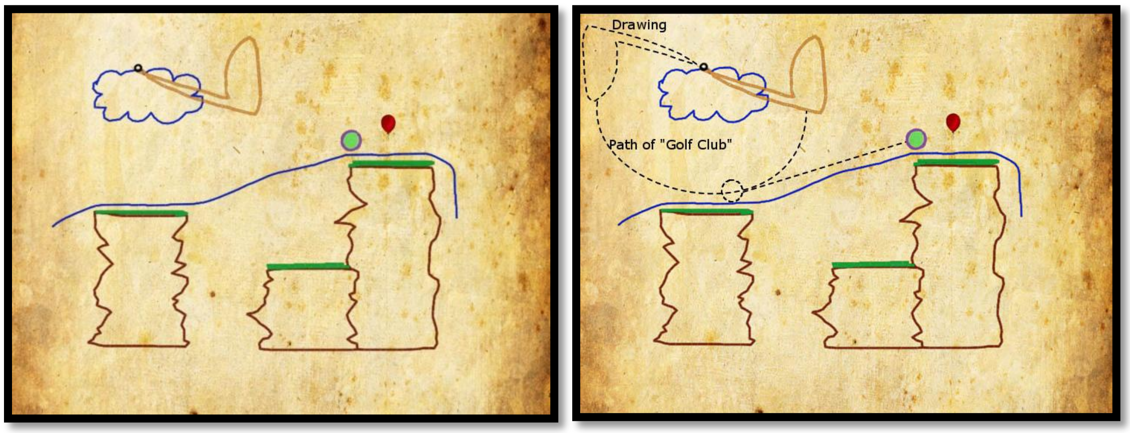


Figure 13-1. Golf problem in NP (left is solution; right is path of motion)

All solutions are drawn with colored markers using the mouse. In a number of cases, the ball must go over a pit. If the ball falls into the pit, the player must start the problem over. Players can replay a problem as often as they like – even after successfully solving it. One motivation to replay a problem is to find even more elegant and creative solutions than were generated before. It is not uncommon for a player to revisit/replay particularly challenging problems multiple times, striving for a better, more elegant solution.

Our system for agent identification (i.e., detecting the creation and use of simple machines) ignores visual data and instead uses information from the underlying physics simulation to classify agents of force and motion. We identify a primary object (PO) for each agent that provides the most salient features for the identification of that agent. That is, the ramp and pendulum agents each use only one object, thus it is the PO. For a springboard, the PO is the one that “springs” up to propel the ball. Finally, the PO in the lever agent is the object that rotates under the load of another object, to lift the ball. Our method continuously monitors all objects in the game for telltale characteristics of the PO in each particular agent of force and motion.

We divide the agent identification process into three stages: *default*, *monitor*, and *identify*. All objects start in the *default* phase for each agent. When an object exhibits characteristics of the PO for any particular agent, it is elevated to the *monitor* stage. Once an object is in the *monitor* stage for a particular agent, detailed data about its movement and interactions with the game world are recorded and the object will inevitably move on to the *identify* stage, where the gathered data are analyzed and a decision is made about whether it is indeed the primary object in a current manifestation of the corresponding agent of motion.

To illustrate our agent identification system, we now describe the process of identifying the pendulum agent of motion (which is the agent used in Figure 13-1 to solve the golf problem). A drawn object begins in the default stage for the pendulum agent. When the object meets the following criteria it is elevated to the monitor stage: (1) the object is attached to a single pin, and (2) the object has rotated more than 20°. The monitor stage will gather physics data for ¾ second and then the agent identification stage will be triggered. A positive identification is made if, during the monitor phase, the object made contact with the ball (i.e., pendulum strike) and the ball moved more than a preset distance. Regardless of whether the identification is made, the object is then lowered back down to the default stage. The classification of other agents occurs in a similar manner.

Task Modeling for Informal Physics

All NP problems require the player to use one or more agents of force and motion in the solution. Successful solutions thus inform one or more of the competencies that we hope to develop in the student. As an illustration, consider the problem called *ballistic pendulum*, shown in Figure 13-2.

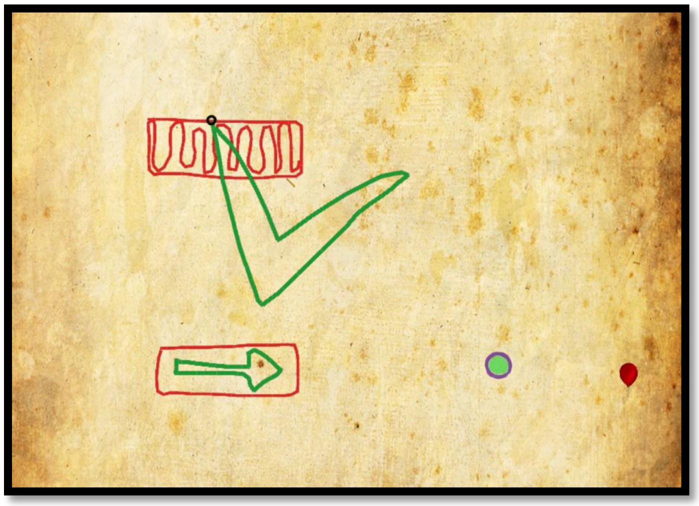


Figure 13-2. Ballistic pendulum problem

This problem requires the student to create a pendulum shape with sufficient mass and positioning so that the pendulum will fall down and “kick” the ball into a free-fall trajectory that ends up landing on the red balloon (the figure shows the ball en route to the balloon). Successfully solving this problem suggests that the student has an intuitive grasp of the concepts of torque, linear, and angular momentum since the correct application of each is required to get the ball to the balloon. Incidentally, the ballistic pendulum is also an experiment often done in introductory physics courses in high school or college.

Other Gameplay Features

NP consists of 7 playgrounds (each one containing around 10–11 levels) that progressively get more difficult. The difficulty of a problem is dependent on a number of factors including: relative location of ball to balloon, obstacles, number of agents required to solve the problem, and novelty of the problem. NP also has introductory videos that show how to use the various agents of force and motion. These tutorials illustrate how to draw each agent to solve a simple problem (during gameplay, students have the option to watch any agent-drawing video at any time).

Object Limit

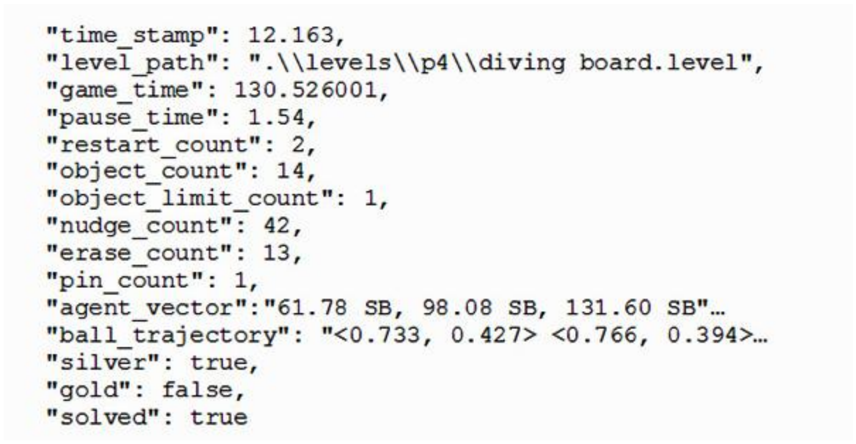
In pilot testing, we discovered that players sometimes opt to draw lines under the ball in order to move it (called line stacking). Repeated use of line stacking can lead players to not learn the agents of force and motion. In order to preclude line stacking to obtain a solution we implemented an "object limit" of 10 objects per level. Once a player draws 10 lines no more lines can be drawn until a line is deleted or the problem is reset (by hitting the space bar). Students can see their line limit in the bottom right hand side of the screen.

Trophies

NP also displays silver and gold trophies in the top left part of the screen, which represent progress in the game. A silver trophy is obtained for any solution to a problem. Players receive a gold trophy if their solution is under a certain number of objects (the threshold varies by problem, but is usually less than three objects). So a player can receive a silver and gold solution for each problem. It is not uncommon for a player to revisit/replay particularly challenging problems multiple times to receive a gold solution.

Log Files

The heart of the stealth assessment lies in the log files generated by NP. NP automatically uploads log files to a server for a session (i.e., log activity between login and log out). Figure 13-3 displays what the log file looks like for one problem.



```
"time_stamp": 12.163,
"level_path": ".\\levels\\p4\\diving board.level",
"game_time": 130.526001,
"pause_time": 1.54,
"restart_count": 2,
"object_count": 14,
"object_limit_count": 1,
"nudge_count": 42,
"erase_count": 13,
"pin_count": 1,
"agent_vector":"61.78 SB, 98.08 SB, 131.60 SB"…
"ball_trajectory": "<0.733, 0.427> <0.766, 0.394>…
"silver": true,
"gold": false,
"solved": true
```

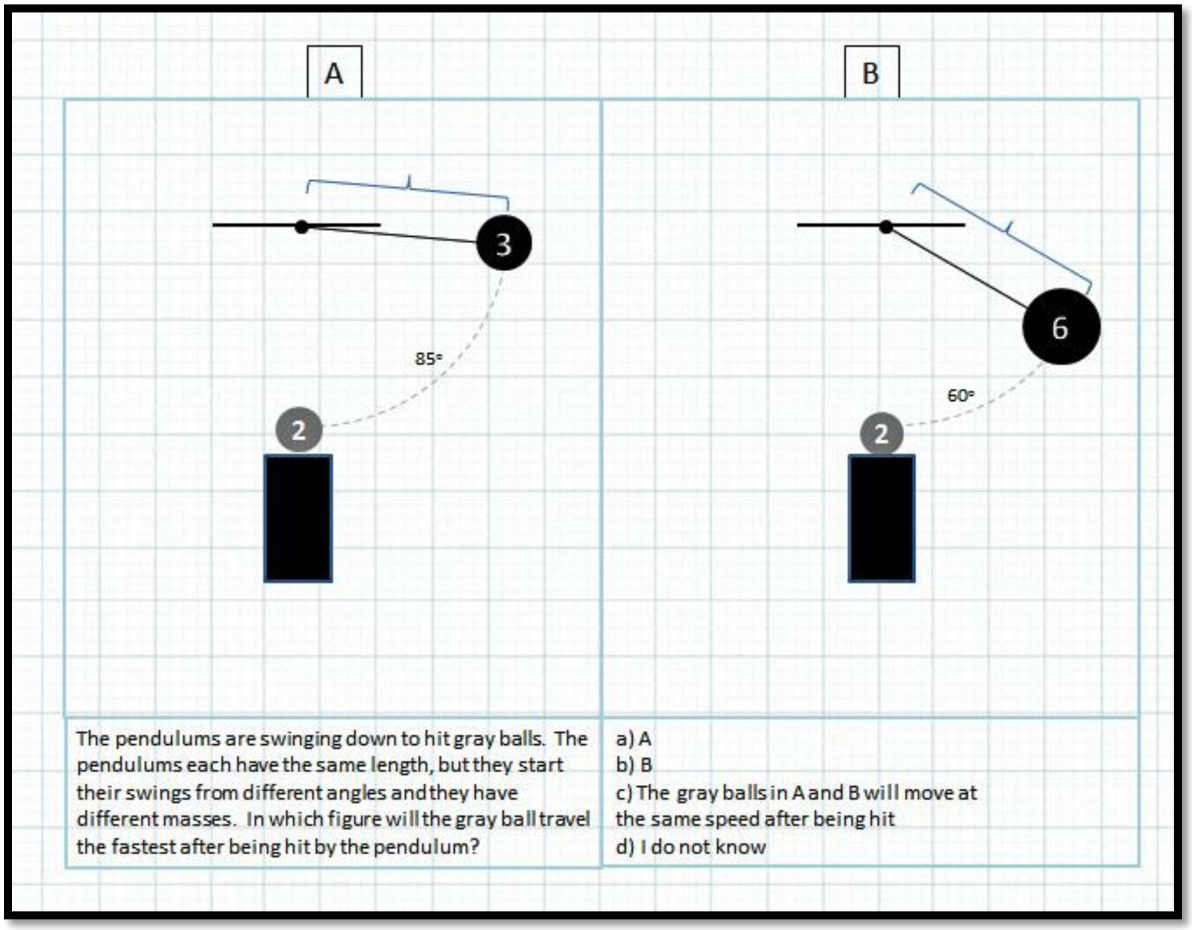
Figure 13-3. Example log file

The session log displays counts and times for several features of gameplay relevant to physics. For example, the "object limit count" reports the number of time a player exceeds the object limit, which can be seen as a lack of knowledge of a particular agent of motion (depending on the problem). Also the "agent vector" reports the agents used in the problem along with the time stamp it was executed (e.g., at timestamp 61.78, a springboard [SB] was created). Finally, the "ball trajectory" reports the 2-D coordinates of the ball over the last few seconds of a solved problem (i.e., the "solution path" of the ball).

A second log file NP reports is called a "replay file." The replay file records all player interactions with the game while attempting to solve a problem. Such interactions include drawing and erasing game objects, creating pins and nudging the ball. NP can read this file to render a visual replay of a problem attempt in real time. The replay system was integral in tuning and verifying the accuracy of the automatic agent identification system.

Preliminary Results

We recently conducted a study where we had middle school students (*n* = 165) play NP for around 5 hours (split into six 45-min sessions). Working with a physics professor, we developed a physics test that assesses informal physics knowledge and does not require math for solutions to physics problems. For example, Figure 13-4 shows an item involving a pendulum. The correct answer is "B." We administered the informal physics pretest at the beginning and a post-test at the end of the gameplay sessions.



The pendulums are swinging down to hit gray balls. The pendulums each have the same length, but they start their swings from different angles and they have different masses. In which figure will the gray ball travel the fastest after being hit by the pendulum?

a) A
b) B
c) The gray balls in A and B will move at the same speed after being hit
d) I do not know

Figure 13-4. Example item from informal physics test

So far, we have found a significant difference between the pre-test and post-test scores (t (154) = 2.12, $p < 0.05$). Students playing the game improved in their informal, conceptual physics understanding over time. Current analyses are revealing that our stealth assessment indicators correlate to a player's knowledge of physics concepts as measured by the tests.

We now turn our attention to how stealth assessment may be employed within the GIFT architecture.

Stealth Assessment within GIFT

With domain-independency being a major requirement in the development of GIFT, it is important that the architecture supports varying open and dynamic game-based learning environments that apply distinctively different messaging protocols. This involves embedding components and processes within GIFT's domain module to support the authoring of stealth assessments regardless of the game-engine being used. Rules and models built around game interaction must be explicitly linked to concepts defined inside the GIFT architecture. This enables the assessment of concepts within GIFT's domain schema as it relates to evidence captured in a game. Based around this notion, GIFT must support the linkage of activities in a game with defined objectives that denote competent behavior within a given domain, regardless of the data structure being extracted from the game environment. For this purpose, a "Gateway Module" is incorporated that associates an external educational/training system's state data with a domain or competency model built within GIFT. This linkage allows for two disparate systems to communicate with one another. In the case of GIFT, this enables the application of AI tools and methods that facilitate real-time assessment of player actions as they relate to desired learning and performance processes and outcomes.

The challenge is that a majority of gaming platforms apply different messaging protocols. Developing approaches to rapidly pair GIFT with any platform is recommended, which will increase its utility across multiple learning environments. This allows any system to link interaction with GIFT's domain model, where assessments are conducted and progress is communicated to the learner model for determining transitions in performance or competency. This approach to assessment is ideal in game-based environments as tracking interaction data as they relate to objectives can denote comprehension and understanding that is difficult to gauge in traditional assessment techniques. The application of stealth assessment within GIFT potentially provides further diagnosis of game performance, which can be communicated to the pedagogical model for more focused selection of feedback and remediation tactics.

As discussed earlier, stealth assessment is dependent on data streams that can be pulled out of the system. In the example of NP, player interaction is monitored for the purpose of capturing a player's creation and use of the relevant agents and inferring how their actions relate to mastery of informal physics knowledge within a specified problem space. Here, user interaction is monitored to determine the application of agents for solving a problem.

Within the context of game-based systems, the domain model in GIFT must accommodate the inclusion of stealth assessment techniques, such as those implemented in NP, that distinguish competency from interaction within a dynamic environment where learners have free control of their movements. These relationships are currently authored in GIFT's Domain Knowledge File (DKF), where a structured XML schema is used to associate specific domain content with generalized tags that can be communicated to the learner model (i.e., concepts, objects, and assessment logic). In the initial releases of GIFT, each of these components are hand-coded by programmers as they link available gameplay data with defined concepts as they relate to the objectives being instructed. This is not ideal, as the goal of GIFT is to enable instructors and trainers to author intelligent tutoring capabilities without possessing skills across the multiple disciplines (e.g., computer science, instructional design, psychometrics, cognitive psychology) required to build such a system.

To ease this burden, research is necessary to identify and develop tools that intuitively guide a course developer through the authoring process. Ultimately, the overarching goal is for this process to involve minimal to no programming by using natural language and well-developed user interfaces to express policies that can be converted to code and implemented for real-time application. One such emerging technology is InternationalTechnology Alliance (ITA) Controlled English, which is a controlled natural language that is unambiguous for computers and allows for the definition and expression of concepts, rules, and relationships (Harries, Braines & Gibson, 2012). Another available tool is Engineering and Computer Simulations' (ECS) Student Information Models for Intelligent Learning Environments (SIMILE), which uses a set of standards-based data models and protocols that associate events within a game to learning objectives, tasks conditions, and standards for a given lesson/scenario. Researchers should explore the application of these tools and others like it for authoring assessments as they relate to particular activities taken within a gaming environment.

In the current version of GIFT, there is a use case within Virtual Battle Space 2 (VBS2), a game-engine used by the Department of Defense for training purposes. In this instance, assessments are based around Distributed Interactive Simulation (DIS) Protocol Data Units (PDUs) that provide entity and environment state data information, and how they relate to defined objects (e.g., waypoints and time sequences). An example is determining what entrance a player used to enter a building, with waypoints placed at each entrance. When a player crosses one of these objects, the system can determine exactly which doorway a player used and this is then passed to GIFT for assessment purposes. If the intention is for a player to select a specific entrance, formative feedback can be triggered when a non-optimal doorway is selected. Currently, each defined object is identified through the VBS2 mission editor and then translated into the DKF. Research should be conducted looking at approaches to link game mission/scenario editors with the associated DKF. This would enable autonomous populating of fields within the XML schema based on defining attributes among available elements within the application. As related to the building entrance example above, an author could identify a location on a scenario map as an object for use in GIFT assessment, rather than the individual having to pull the data and entering them manually into the DKF.

Another area of interest in game-based intelligent tutoring is being able to predict performance as a player progresses through a scenario so that interventions can be applied before user actions lead to poor performance outcomes. Research related to this function is the application of Markov decision processes (MDPs) that map associated goal objectives to state spaces in a game. Current applications include reinforcement learning and partially observable MDPs (Sigaud & Buffet, 2010; Folsom-Kovarik, Sukthankar & Schatz, in press). MDPs can be applied to determine if a player is heading down a non-optimal path as it relates to defined standards, and can be used to deem what goals are being valued as it relates to meeting scenario objectives. It is recommended that researchers look into how MDPs can be used in GIFT for predicting performance outcomes and providing diagnostics into what elements of their interaction are causing the decrements in performance.

Discussion and Future Research

Well-designed games can provide meaningful assessment environments by providing players with problems that require the application of various competencies, then monitoring their performance. In this chapter, we presented an assessment methodology that enables us to develop tasks in digital games designed to elicit specific performance data that are then statistically linked to our focal competencies.

The research can expand in a number of general directions. That is, we can push the bounds of our stealth assessments relative to implementing the models in additional digital games as well as other digital learning environments to determine the range of environments that may employ the same competency and evidence models for scalable, cost-effective, and engaging solutions to the assessment of complex competencies. In addition, we can examine any added value of including exploratory, data-mining methods to stealth assessment's more theoretically driven approach regarding the quality of the assessment.

As we consider the development of stealth assessments for a diverse range of games and other learning environments, the need for an interface to communicate equally diverse data to GIFT's domain module becomes apparent. As we alluded to in the previous section, the interface must be portable and flexible, both in terms of technology and content. Thus, it is important to research elements that are common to stealth assessments across learning environments and areas of assessment. The evidence-based models we developed for NP is a good start.

Regarding future research related to learning, stealth assessment has the potential to be quite useful for diagnostic purposes due to the fine-grained analysis of student behavior in situated contexts. In addition, real-time information about player competency states can be useful to support learning through hints and feedback, as well as dynamic matching of game difficulty level to player ability (e.g., providing more challenging problems for those with high levels of various skills). Regarding the example used in this chapter, the indicators linked to the agents of force and motion can serve as the basis for diagnoses. For instance, if a student created a lever that did not successfully solve a problem that could have been solved with a lever, the indicators would inform the most likely reason(s) why. That is, the lever may have failed given (1) the wrong mass of an object that was used on one side of the lever, (2) the fulcrum was positioned inaccurately, and/or (3) the size/length of the lever was too short or too long. Those data (mass, position, and length) are calculated as part of the stealth assessment.

We are excited that researchers are starting to use digital games for learning and assessment. We think stealth assessment is one way to maximize the positive impact digital games can have on students. As a result, the future developmental efforts of GIFT should aim at identifying authoring tools and methods that ease the process of embedding stealth assessment capabilities in game-based learning environments.

Acknowledgements

We would like to sincerely thank the Bill and Melinda Gates Foundation for funding the stealth assessment project described herein.

References

Arum R. & Roska, J. (2011). *Academically adrift: Limited learning on college campuses*. Chicago, IL: The University of Chicago Press.

Bridgeland, J. M., DiIulio, J. J., Jr. & Morison, K. B. (2006). *The silent epidemic: Perspectives of high school dropouts.* Washington, DC: Civic Enterprises and Peter D. Hart Research Associates.

Caramazza, A., McCloskey, M. & Green, B. (1981). Naive beliefs in "sophisticated" subjects: Misconceptions about trajectories of objects. *Cognition, 9,* 117-123.

Crouch, C. A. & Mazur, E. (2001) Peer instruction: Ten years of experience and results. *American Journal of Physics, 69*, 970-977.

Dede, C. (2005). Planning for neomillennial learning styles. *EDUCAUSE Quarterly, 28*(1), 7-12.

DiCerbo, K. E. & Behrens, J. T. (2012). Implications of the digital ocean on current and future assessment. In R. Lissitz & H. Jiao (Eds.) *Computers and their impact on state assessment: Recent history and predictions for the futur*e (pp. 273-306). Charlotte, NC : Information Age Publishing.

diSessa, A. A. (1982). Unlearning Aristotelian physics: A study of knowledge-based learning. *Cognitive Science, 6,* 37-75.

Feynman, R. P., Leighton, R. B. & Sands, M. (1964). *The Feynman lectures in physics*. Addison Wesley.

Feynman, R. P. (1964). *The character of physical law*, Cornell, NY: University Press.

Folsom-Kovarik, J.T., Sukthankar, G. & Schatz, S. (in press). Tractable POMDP Representations for Intelligent Tutoring Systems. *ACM Transactions on Intelligent Systems and Technologies.*

Gee, J. P., Hull, G. A. & Lankshear, C. (1996). *The new work order: Behind the language of the new capitalism.* St. Leonards Australia: Allen & Unwin.

Gronhaug, K. & Kaufman. G. (Eds.) (1988). *Innovation: A cross-disciplinary perspective*. Oslo, Norway: Norwegian Universities Press/Oxford University Press.

Hake, R. R. (1998). Interactive engagement vs. traditional methods in mechanics instruction. *American Journal of Physics, 66*(1), 64-74.

Halloun, I. (1996) Schematic modeling for meaningful learning of physics, *Journal of Research in Science Teaching, 33,* 407-431.

Halloun, I. & Hestenes, D. (1985). Initial knowledge state of college physics students, *American Journal of Physics, 53,* 1043-1055.

Harries, D., Braines, D. & Gibson, C. (2012). Towards an expression of Policy in Controlled English. *In the Proceedings of the 6th Annual Conference of the International Technology Alliance*. Botley, UK.

Hestenes, D. & Wells, M. (1992). A mechanics baseline test. *The Physics Teacher, 30*(3), 159-167.

Hestenes, D., Wells, M. & Swackhamer, G. (1992). Force Concept Inventory. *The Physics Teacher, 30,* 141-151.

Hewitt, P. G. (2009). Conceptual physics (11th ed.). San Francisco, CA: Pearson Education.

Hiebert, E. H., Valencia, S. W. & Afflerbach, P. P. (1994). Understand authentic reading assessment: Definitions and perspectives. In S. W. Valencia, E. H. Hiebert & P. P. Afflerbach (Eds.), *Authentic reading assessment: Practices and possibilities* (pp. 6-21). Newark, DE: International Reading Association.

Kellaghan, T. & Madaus, G. F. (1991). National testing: Lessons for America from Europe. *Educational Leadership, 49*(3), 87-93.

Lenhart, A., Kahne, J., Middaugh, E., Macgill, A. R., Evans, C. & Vitak, J. (2008). *Teens' gaming experiences are diverse and include significant social interaction and civic engagement*. Washington, DC: Pew Internet & American Life Project.

Madaus, G. & O'Dwyer, L. (1999). A short history of performance assessment. *Phi Delta Kappan, 80*(9), 688-695.

Masson, M. E. J., Bub, D. N. & Lalonde, C. E. (2011). Video-game training and naive reasoning about object motion. *Applied Cognitive Psychology, 25,* 166-173.

McCloskey, M. & Kohl, D. (1983). Naive physics: The curvilinear impetus principle and its role in interactions with moving objects. Journal of Experimental Psychology: *Learning, Memory & Cognition, 9,* 146-156.

McDermott, L. (1993). How we teach and how students learn – A mismatch? *American Journal of Physics, 61,* 295-298.

Quellmalz, E.S., Timms, M. J., Buckley, B. C., Silberglitt, M. & Brenner, D. (2012) SimScientists: Measurement in simulation-based science assessments Manuscript submitted for publication.

Reiner, C., Proffit, D. R. & Salthouse, T. (2005). A psychometric approach to intuitive physics. *Psychonomic Bulletin and Review, 12,* 740–745.

Shepard, L.A. (1991). Will national tests improve student learning? *Phi Delta Kappan, 72,* 232-238. Retrieved from: www.cse.ucla.edu/products/Reports/TECH342.pdf

Shute, V. J. (2011). Stealth assessment in computer-based games to support learning. In S. Tobias & J. D. Fletcher (Eds.), *Computer games and instruction* (pp. 503-524). Charlotte, NC: Information Age Publishers.

Shute, V. J. & Ke, F. (2012). Games, learning, and assessment. In D. Ifenthaler, D. Eseryel & Ge, X. (Eds.), *Assessment in game-based learning: Foundations, innovations, and perspectives* (pp. 43-58). New York, NY: Springer.

Shute, V. J. & Ventura, M. (2013). Measuring and supporting learning in games: Stealth assessment. Cambridge, MA: The MIT Press.

Sigaud, O. & Buffet, O. (2010). *Markov Decision Processes in Artificial Intelligence*. Wiley-IEEE Press.

Swann, W. F. G. (1950). The teaching of physics. *American Journal of Physics, 19*(2), 182-187.

Tobias, S. & Fletcher, J. D. (Eds.) (2011). *Computer games and instruction*. Charlotte, NC: Information Age Publishers.

Tobias, S., Fletcher, J. D., Dai, D. Y. & Wind, A. P. (2011). Review of research on computer games. In S. Tobias & J. D. Fletcher (Eds.), *Computer games and instruction* (pp. 127-222). Charlotte, NC: Information Age.

White, B. Y. (1994). Designing computer games to help physics students understand Newton's laws of motion. *Cognition and Instruction, 1*(1), 69-108.

Wilson, K. A., Bedwell, W., Lazzara, E. H., Salas, E., Burke, C. S., Estock, J., … Conkey, C. (2009). Relationships between game attributes and learning outcomes: Review and research proposals. *Simulation & Gaming, 40*(2), 217-266.

CHAPTER 14 –Assessing the Disengaged Behaviors of Learners

Ryan S.J.d. Baker[1], Lisa M. Rossi[2]
[1]Columbia University Teachers College, [2]Worcester Polytechnic Institute

Introduction

In recent years, an increasing number of models have been published that can infer if a learner is behaviorally disengaged while working within an interactive learning environment, and can conduct inference using features of data focused on learner interaction with the learning system. In this chapter, we discuss some of the behaviors that have been shown to be amenable to this type of modeling, including off-task behavior, gaming the system, and carelessness. We also consider some of the algorithms and approaches that have been found to be particularly effective. We contemplate the relative merits of knowledge engineering and data-mining approaches for this type of model, and focus on the key validity concerns that must be addressed for these types of models to be used with confidence in a comprehensive framework such as GIFT.

Gaps

Over the last decades, adaptive computerized instruction has become increasingly effective at assessing the knowledge state of a learner (Corbett & Anderson, 1995; Martin & VanLehn, 1995; Shute, 1995; Pavlik et al., 2009; Pardos et al., 2011), supporting automated decisions about which content to assign to students through the implementation of strategies such as mastery learning, where a student is assigned content for a specific skill or knowledge component until demonstrating mastery (Corbett, 2001).

However, despite recent advances in the assessment of student disengagement (discussed in this chapter), and a small number of successful cases of models of disengaged behavior being used in learning interventions (Baker et al., 2006; Walonoski & Heffernan, 2006; Arroyo et al., 2007), adaptive computerized instruction is generally not as adaptive to engagement as it is to student knowledge.

There are likely multiple reasons for this. First, engagement models thus far have had to be created for specific learning environments, with only moderate similarity for models of the same construct created for different learning environment. By contrast, creating knowledge models often involves applying one of a small set of known algorithms to a data set in a standard fashion (Pardos et al., 2011). This often requires some knowledge engineering and rational modeling to create mappings between items and cognitive skills, but that process is relatively well known and has been conducted for a large range of learning environments. Second, validating models of engagement is more challenging than validating knowledge models; knowledge models are often validated in terms of whether predictions of future student behavior are correct (Corbett & Anderson, 1995; Pavlik et al., 2009; Pardos et al., 2011), but validating an engagement model beyond face validity requires collecting human labels of data in terms of the target construct, typically involving external coders (Baker et al., 2004; Baker, 2007b; Cetintas et al., 2010; Walonoski & Heffernan, 2006; Wixon et al., 2012). These labels often must be collected at considerable scale to ensure generalizability to a large and diverse target population.

Emerging Concept, Model, or Method

One of the challenges with modeling engagement within the context of adaptive computerized instruction is deciding which dimension(s) of engagement to model. Fredricks, Blumenfeld, and Paris (2004) have proposed that engagement be studied as a multifaceted construct, with behavioral, affective, and cognitive dimensions. These dimensions can be understood as follows: behavioral engagement centers on the action of participation in an educational interaction (including academic, social, and extracurricular activities), affective engagement focuses on both positive and negative emotional reactions (with regard to teachers, peers, or academics), and cognitive engagement is based on investment at a cognitive level and thoughtfulness. Within these dimensions, there are many constructs, e.g., many behaviors that indicate engagement or disengagement. Each of these constructs can also be defined in a range of ways. This multifaceted view of engagement imposes added complexity to our ability to infer disengagement, as it broadens the breadth of behaviors necessary to detect in order to achieve a full multidimensional picture of a learner's engagement. Fortunately, however, not all dimensions (or aspects of each dimension) need to be detected in order to support effective intervention. In addition, specific behaviors impact learning outcomes and longer-term engagement in different ways, and some are more important to identify and adapt to than others, depending on the learning context.

Despite these complexities, opportunities exist for a combination of rich logs of student interaction and EDM methods to be used in concert to create richer detectors of student disengagement with regard to all three facets of the construct.

In this chapter, we look at work that models the behavioral dimensions of engagement, focusing on work to identify behavioral disengagement in the types of adaptive computerized instruction being integrated with the GIFT framework (such as intelligent tutors, simulations, and serious games). We discuss the following behaviors: off-task behavior, gaming the system, carelessness, and WTF behavior (Wixon, Baker, Gobert, Ocumpaugh & Bachmann, 2012), which have been detected with validated models operating on data from learner interactions with adaptive computerized instruction (e.g., no physical sensors).

Modeling Off-Task Behavior

The first automated detector of whether a student is off-task within adaptive computerized instruction was published in Baker (2007b), which presented a machine-learned model to detect off-task behavior of students using an ITS for middle-school mathematics. Off-task behaviors were defined as behaviors that do not involve the system or learning task (including off-task conversation, off-task solitary behavior, and inactivity), building off past work studying off-task behavior within traditional classroom settings (Karweit & Slavin, 1982; Lahaderne, 1968; Lee, Kelly, and Nyre, 1999). These models were built using data from quantitative field observations conducted in middle-school classrooms by trained field coders, as training data. Models were validated by conducting cross-validation at the student level. Data features included in the model were the following:

- Details of student actions, such as whether the action was correct and the type of user interaction used in the current problem step (e.g., choosing from a set of options, inputting an answer, or plotting points; some types of interactions naturally take longer than others).

- Whether it was the student's first attempt at the problem.

- Time taken to complete a problem, expressed in three ways: (1) how many seconds the action took, (2) how many standard deviations faster or slower the action took compared to other students, and (3) time taken in the last three or five actions expressed as the sum of standard deviations faster or slower than other students.

In order to model off-task behavior, Latent Response Models (LRM) were used as the statistical basis for all detectors in this study, as they are able to easily and naturally integrate multiple data sources at different grain sizes. This framework included one observable level (assessing how frequently each student is off-task) and two hidden (latent) levels. The detector determines the proportion of time spent off-task by making a binary assessment as to whether each individual student action is off-task and determining the percentage of actions which are assessed to be off-task for each student. Model selection for the multiple-feature models were validated by adding parameters to the model until the parameter worsened the model's performance in a student-level tenfold cross-validation. It was found that the best-fitting multiple-parameter model fit the data with a cross-validated correlation of 0.55.

A second automated detector of whether a student is off-task was developed by Cetintas and colleagues (2010), who added data on mouse movement to the features used in Baker (2007b), modeling student off-task behavior within a math tutoring program designed to help elementary school students with learning disabilities and/or emotional disorders learn problem solving skills for Equal Group and Multiplicative Compare problems. This more multimodal approach extended past work that had not considered mouse movement data. Their approach also includes personalization of the detector to account for inter-user variability of behavior. The personalized version of the model incorporates data from a student's past trials on each problem, which are used to generate a student-specific version of each feature while predicting that student's behaviors for the current trial. The same general approach to coding students as on-task or off-task as was used in Baker (2007b) was used in this work. Quantitative field observations were conducted in the classrooms by trained field coders and synchronized with the log data. Students were observed sequentially (to avoid observer bias) and were coded as off-task if they were observed doing any of the following for more than 30 seconds: talking with another student about anything other than the subject material, being inactive, or exhibiting off-task solitary behavior.

Within this work, ridge regression (Hoerl & Kennard, 1970) was used to estimate model parameters, and it was found that this approach (including mouse movement data and student-specific models) led to better detection of off-task behavior than approaches lacking one or more of these features.

Modeling Gaming the System

Gaming the system is defined as attempting to succeed in an education environment by exploiting properties of the system rather than by learning the material and trying to use that knowledge to answer correctly (Baker et al., 2006). The first automated detectors of gaming the system were presented by Aleven et al. (2004) and Baker, Corbett, and Koedinger (2004).

One category of gaming detectors was developed using knowledge engineering (Beal, Qu & Lee, 2006; Buckley, Gobert & Horwitz, 2006; Shih, Koedinger & Scheines, 2008), where rational analysis is applied by a human analyst to derive a model that can be applied. Within knowledge engineering approaches, there is typically no gold standard used to validate models; models and their parameters are generated based on the judgment of the researcher(s), although in some cases research models are compared to learning outcomes. The first such detector was presented in Aleven et al. (2004) and refined within Aleven et al. (2006). This detector was developed using a knowledge engineering approach and took the form of a set of simple production rules. The detector was initially developed in the context of data from a Cognitive Tutor on geometry, but has since been applied to other intelligent tutors. Within this model, gaming behaviors were defined by a pair of rules, *Clicking Through Hints*, which consists of requesting a hint, and then requesting another hint too rapidly to read the first hint (defined as under 5 seconds), and *Try-Step Abuse,* where a student response took under 7 seconds. In related work, Beck (2005) looked at quick responses on difficult items, parameterizing "quick" and "difficult" and fitting values for these parameters based on data. Similar knowledge engineered models have been presented by Beal et al. (2006), Johns & Woolf (2006), Gong, Beck, and Heffernan (2010), and Muldner et al. (2011).

A second category of gaming detectors was developed using a combination of human labels of gaming behavior and data mining/machine learning methods, where a model is trained to infer what the human coder's labels are. The first such detector was presented in Baker, Corbett, and Koedinger (2004) and refined within Baker et al. (2008). This detector identified gaming the system and distinguished between gaming behaviors characterizing unsuccessful students and gaming behaviors characterizing more successful students (the primary distinction was in terms of when gaming occurs, with unsuccessful students gaming on difficult material and successful students gaming on easier material). This detector was built for students using a Cognitive Tutor for middle school, using quantitative field observations and tutor log data collected across a total of four tutor lessons used by over 400 students in two separate school districts. Data features used in the model included time expressed in terms of how many seconds the action took, how many standard deviations faster or slower the action took compared to other students, and time taken in the last three or five actions expressed as the sum of standard deviations faster or slower than other students. Details about the interaction were also included regarding the learning system's assessment of the action (as correct; incorrect, indicating a known bug; or a help request), the type of interface widget involved in the action, and whether this attempt was the student's first attempt. The automated detectors, developed using a LRM framework (Maris, 1995) integrated the field observations and tutor logs, at different grain sizes, into a single model. In a gaming detector's outermost/observable layer, the gaming detector assessed how frequently each of *n* students is gaming the system. The gaming detector's assessments for each student were then compared to the observed proportions of time each student spent gaming the system. The detector was validated to be able to generalize to new students and new tutor lessons.

In later work to detect gaming using machine-learning and human labels, Walonoski and Heffernan (2006) improved on the very coarse-grained label synchronization in Baker's work by using time windows of five different sizes (30 seconds, and 1, 2, 4, and 6 min), increasing the degree to which the model was fine-grained. Next, work by Baker and de Carvalho (2008) achieved 20-second-level synchronization by using text replays rather than quantitative field observations, for human labeling. Text replays are log files presented in textual form, and represent a segment of student behavior during a pre-selected duration of time or length. It has been shown that text replays have good inter-rater reliability and agree well with prediction made by models generated using quantitative field observation data (Baker, Corbett & Wagner, 2006). Using labeling at this label made it possible to use off the shelf classifiers, in this case J48 Decision Trees, an open-source implementation of C4.5 Decision Trees (Quinlan, 1993). In further work, this approach was extended to a constraint-based tutor for database programming (Baker, Mitrovic & Mathews, 2010) and a handheld app was developed to support synchronization of field observations with similar precision to what can be achieved with text replays (Ocumpaugh et al., 2012).

Modeling Carelessness

The construct of carelessness has been defined in two ways: as an error made on a task that the student already knows (Clements, 1982), or as impulsive and/or hurried actions (Maydeu-Olivares & D'Zurilla, 1995). San Pedro and colleagues (2011a) identified Clements's definition of carelessness as being the same as the contextual probability of slipping on a problem or problem step in an intelligent tutor, a construct that it was previously shown can be inferred through manipulating a BKT model (Baker, Corbett & Aleven, 2008). Based on this theoretical link, San Pedro and colleagues (2011a) manipulated the internal structure of a BKT algorithm in order to develop a model of carelessness, doing so using log files produced within a Cognitive Tutor for Scatterplots. The model of carelessness was developed by first obtaining ground-truth labels using future knowledge to drive a machine-learned model that can predict careless errors without using future data. Then, a model that only uses data from the past was created using sixfold student-level cross-validation linear regression modeling. Creating this model also serves a function of smoothing extreme estimates. The model achieved A' and BIC' values, which indicated that the detector performed better than chance. The same approach was replicated within Science ASSISTments, a set of scientific simulations that scaffold student inquiry processes and assess students' inquiry skill (Hershkovitz et al., 2013).

This model has been validated to work for new students (Baker, Corbett & Aleven, 2008), new populations in different countries (San Pedro et al., 2011a), and additional tutoring systems (Hershkovitz et al., 2011). It has been also been shown to predict student post-test score even when controlling for student knowledge (Baker et al., 2010).

Modeling WTF Behavior/Off-Task Behavior within environment

Rowe and colleagues (2009) reported that students sometimes engage in behavior within online learning that seems unrelated to the student's learning task, giving the example of students climbing on top of (in-game) buildings or putting (in-game) bananas in an (in-game) toilet. They identified this construct as off-task behavior within the learning environment. In 2012, Wixon and colleagues (2012) argued that this term may obscure considerable differences in why students engage in this behavior compared to traditional off-task behavior (where the student ceases to work on the task at all), as well as differences in impact, and suggested the alternate term of WTF behavior.

Rowe and colleagues proposed an operational definition as behaviors that are clearly unrelated to the narrative and curriculum, and built a knowledge-engineered model to infer this construct in a narrative-centered learning environment called Crystal Island, in which students solve scientific puzzles presented through interactive story scenarios. Their definition when expanded by Sabourin et al. (2011) consisted of: interactions with in-game objects that are not relevant to the scientific puzzle, moving a task-related object to an unrelated location, spending too much time in an irrelevant location, or exceeding a height achievable by normal navigation.

Wixon and colleagues (2012) developed a data-mined automated detector of WTF behavior, within the context of Science ASSISTments. Within this environment, WTF behavior involves behaviors such as changing variable values many times without running trials, or rapidly pausing and unpausing a simulation. Ground-truth labels for this behavior were developed using text replays, and then a set of features were distilled using code that had been previously developed to detect student use of experimentation strategies and hypothesis testing within Science ASSISTments. Eleven common classification algorithms were attempted to fit detectors of WTF, and the best model performance was achieved by the Projective Adaptive Resonance Theory Model (PART) algorithm (Frank & Witten, 1998), which produces rules out of C4.5/J48 decision trees. The models were evaluated using a process of sixfold student-level cross-validation, and the detectors were assessed using four metrics: A', Kappa, precision, and recall.

Use of Detectors in Intervention and "Discovery with Models" Analyses

Once detectors of disengaged behavior have been developed, they can be used in two fashions: within "Discovery with Models" analyses to understand the relationship that the disengaged behavior has to other constructs, and within interventions, by embedding the detectors in running software to drive adaptation, and using them to change the system's behavior.

Automated detectors of off-task behavior, gaming the system, carelessness, and WTF behavior have been used in several "discovery with models" analyses. Early analyses on gaming the system indicated that gaming was associated with poorer learning (Baker et al., 2004; Beck, 2005), although fast responses are positively correlated with learning if correctness is not taken into account (e.g., Aleven et al., 2006). This research was followed up by work by Cocea, Hershkovitz, and Baker (2009), who studied whether off-task behavior and gaming the system had an immediate impact on learning or a more aggregate impact on learning, and found evidence that off-task behavior was not associated with worse performance in the short-term, but that it led to the student having fewer opportunities to practice the skill, leading to smaller learning gains over time. By contrast, gaming the system was found to be associated with worse performance in the short-term as well as the long-term. Both Rowe et al. (2009) and Sabourin and colleagues (2011) found evidence that WTF behavior is associated with lower learning gains.

Baker (2007a) used an automated detector of gaming the system to determine whether differences in the frequency of student gaming were better predicted by tutor content than by which student was using the software. Interestingly, knowledge-engineered models have produced the opposite finding, that students were better predictors of gaming behavior than the lesson (Gong et al., 2010; Muldner et al., 2011), a contrasting finding which has not entirely been reconciled, although recent collaborative work between some of the authors of Baker (2007a) and Muldner et al. (2011) suggest that this contrast may be because the different detectors identify different behavior in general. Baker et al. (2009) and Baker (2009) followed up the finding in Baker (2007a) by studying which differences in lesson features predict the degree to which students will go off-task or game the system in a lesson, by combining data from a taxonomy of differences between 22 lessons with assessments of how often a set of students was off-task or gaming in each lesson, across the course of a year. They found that several features were associated with gaming, including ineffective or overly abstract hints, unclear toolbar icons or problem flow, and the lack of (interest-increasing) extraneous text in problem statements.

Detectors of off-task behavior, gaming the system, and carelessness were used in Baker and Gowda (2010) to study the differences in the proportion of these behaviors between students in an urban, rural, and suburban school, across an entire year of usage of a Cognitive Tutor for Geometry. They found that urban school students go off-task and are careless significantly more than rural and suburban school students, and also that gaming the system was most prominent in the urban school. In work within a single population, Rowe and colleagues (2009) found that WTF behavior was significant more common among male students than female students.

San Pedro and colleagues (2011b) used a machine-learned detector of student carelessness to study the relationship between carelessness and affect in high school students using a Cognitive Tutor for Scatterplots. It was found that errors made by students who are confused or bored are less likely to be careless errors. The negative correlation between confusion and carelessness increases in magnitude as students used the tutor more, even as confusion itself decreases in frequency. This suggests that students who were struggling the most and remained confused were less likely to become careless. It was also found that students displaying engaged concentration were more likely to make careless errors, a finding which seems strange but which may be consistent with offline findings in Clements (1982) that successful students are more likely to become careless. Affect was also studied in relation to WTF behavior by Sabourin and colleagues (2011), who investigated the affective role of this category of behavior in Crystal Island. They found that no emotional states were more likely than chance to lead to WTF behavior. However, it was found that students who had remained on-task after reporting confusion were more likely to report feeling focused next, while students who went off-task (in the environment) after reporting confusion were more likely to report boredom or frustration next. It was also found that frustrated students who went off-task (in the environment) were more likely to report feeling focused next, while confused students who went off-task (in the environment) were more likely to report a negative emotion next. By contrast, frustrated students who remained on-task were also more likely to report boredom next. This suggests that this type of behavior may be beneficial to frustrated students by allowing them to distance themselves from the problem.

Research has also been conducted to model the relationship between disengaged behavior and motivational variables. Baker (2007b) found that off-task behavior was associated with disliking computers, disliking mathematics, passive-aggressiveness, and lack of educational self-drive. Baker and Walonoski et al. (2008) investigated which student behaviors, motivations, and emotions are associated with gaming the system, across multiple studies with two different systems. They found that gaming the system was associated with disliking the software's subject matter, lacking self-drive, disliking computers and the learning environment, believing that mathematics ability is innate, and believing that the tutor is not helpful for learning. Beal, Qu, and Lee (2008), using the gaming detectors from Beal, Qu, and Lee (2006), found that students with low math self-concept were most likely to engage in guessing gaming behavior. Hershkovitz and colleagues (2013) studied the relationship between carelessness and motivation within Science ASSISTments, finding carelessness is higher in students characterized by high levels of learning goal orientation and academic efficacy (in the case of academic efficacy, replicating off-line results by Clements [1982]), and high levels of both performance-approach and performance-avoid goals. By contrast, carelessness was lower in students having neither learning nor performance goals. Hershkovitz et al. (2011) also found that students with performance goals demonstrated an increase in carelessness earlier within the set of trials than students with learning goals. On the other hand, students who lacked either type of goal demonstrated consistently higher carelessness over trials.

Detectors of this type have also been used to drive automated interventions, with the goal of improving student engagement and learning. One of the first examples of this can be seen in Baker et al. (2006), where an automated detector of gaming the system was embedded into an interactive agent (similar to non-player characters [NPCs] in games), who displayed negative emotion when students gamed, and who provided supplementary exercises designed to support students in learning material bypassed via gaming. This intervention improved student learning and reducing gaming behavior in the United States (Baker et al., 2006), although its results did not replicate in the Philippines (Rodrigo et al., 2012). Another intervention using gaming detection was developed by Arroyo and colleagues (2007), who provided meta-cognitive messages to students on the negative impact of gaming, combined with visualizations of students' recent gaming behavior. This intervention also reduced gaming and improved learning. Off-task behavior detection, carelessness detection, and WTF behavior detection has not yet been used as the basis of automated intervention, although research projects along these lines are currently underway (Inventado & Numao, 2012).

Discussion, Recommendations, and Future Research

As this chapter indicates, the last several years have seen considerable work in modeling a range of forms of student disengagement, including gaming the system, off-task behavior, WTF behavior, and carelessness. These detectors have been developed through a range of approaches, and a consensus appears to be emerging that disengaged behaviors can be assessed in a range of different online learning environments.

As such, there is an opportunity for a framework such as GIFT to incorporate models of this type. Historically, the ITS field has been better at developing these types of models and using them within discovery with models analyses than it has been in using them to modify tutor pedagogy and adaptivity (although successful examples of the latter exist, particularly for gaming the system – cf. Baker et al., 2006; Walonoski & Heffernan, 2006; Arroyo et al., 2007; Roll et al., 2011). This limitation is holding back the potential of these approaches to provide information that can be used to reengage learners and enhance learning.

A key step that could facilitate incorporation of these types of models into GIFT would be to incorporate tools that support detector-building into the GIFT framework. Several types of tools have been developed to support this process, but the tools have been developed by a variety of research groups and are not well integrated with specific ITSs, or for that matter, with one another. For example, tools have been developed for automated feature generation by a wide variety of research groups, but have been often scoped for use with a single learning environment. One exception is found in the EDM Workbench (Rodrigo et al., 2012), which can generate a range of features for data in the format used by the PSLC DataShop (Koedinger et al., 2010), but can only do so post-hoc. Still, this system's feature generation could be extended and used as a basis for feature generation within the GIFT framework. Ideally, feature generation should be conducted both on existing data sets, and at run time within a tutor, in order to facilitate both the creation and use of automated detectors of disengagement. A tool such as the EDM Workbench could be incorporated into the GIFT framework via creating explicit API-level links between GIFT and the EDM Workbench, where the EDM Workbench could pull data directly from GIFT, and export detectors back to GIFT for use in the user model. In addition, the feature generation code in the EDM Workbench could be integrated into the GIFT framework, so that distilled features could be directly used by detectors exported to GIFT's user model. Simply making these tools available to GIFT users is useful but not sufficient. The process of importing data from GIFT to the EDM Workbench is time consuming if formats do not match, and an unnecessary step compared to the direct approach possible with API-level links. Similarly, the process of manually taking detectors and associated feature distillers from the EDM Workbench and building them into GIFT's user model is challenging if integration is not created between these tools. These issues are not unique to the EDM Workbench, but are general to the problem of using tools for feature distillation or detector building to enhance GIFT. It is worth noting also that tools like the EDM Workbench are useful for modeling a range of behaviors, beyond the disengaged behaviors discussed in this chapter.

A second opportunity is to integrate tools for labeling data in terms of engagement into the GIFT framework. There are currently tools for collecting both text replays (discussed above) and quantitative field observations of disengagement that could be integrated into GIFT. The EDM Workbench offers support for conducting text replays, though in a less visually attractive format than tools designed explicitly for conducting text replays in a single learning system. Tools for generating tailored text replays have also been designed for a variety of learning environments. Integrating text replays into GIFT would be a useful step towards a framework that can incorporate engagement detectors into a wider number of ITSs. Similarly, integrating code into GIFT for conducting field observation of student disengagement, such as the HART app for Android (Ocumpaugh, Baker & Rodrigo, 2012), would support the development of detectors for ITSs using GIFT. Another potential opportunity can be seen in work to develop detectors of carelessness. The development of detectors of carelessness relies upon initial estimates of carelessness computed using student knowledge models. Extending the knowledge modeling in GIFT to produce carelessness labels would be a valuable step towards incorporating this type of adaptation capacity into GIFT.

Through these steps, it will become easier to build automated detectors of student disengagement into the GIFT framework. Doing so will make it feasible to conduct further research on how these models can best be used to reengage learners toward developing understanding in the field as to how ITSs can best adapt to differences in student engagement.

References

Aleven, V., McLaren, B. M., Roll, I. & Koedinger, K. R. (2004). Toward tutoring help seeking: Applying cognitive modeling to meta-cognitive skills. In *Proceedings of the 7th International Conference on Intelligent Tutoring Systems* (pp. 227-239).

Aleven, V., McLaren, B., Roll, I. & Koedinger, K. (2006). Toward meta-cognitive tutoring: A model of help seeking with a Cognitive Tutor. *International Journal of Artificial Intelligence and Education, 16*, 101-128.

Arroyo, I., Ferguson, K., Johns, J., Dragon, T., Meheranian, H., Fisher, D., Barto, A., Mahadevan, S. & Woolf. B. P. (2007). Repairing disengagement with non-invasive interventions. In *Proceedings of the 13th International Conference on Artificial Intelligence in Education* (pp. 195-202).

Baker, R. S. J. d. (2007a). Is gaming the system state-or-trait? Educational data mining through the multi-contextual application of a validated behavioral model. In *Complete On-Line Proceedings of the Workshop on Data Mining for User Modeling at the 11th International Conference on User Modeling 2007* (pp. 76-80).

Baker, R. S. J. d. (2007b). Modeling and understanding students' off-task behavior in intelligent tutoring systems. In *Proceedings of ACM CHI 2007: Computer-Human Interaction* (pp. 1059-1068).

Baker, R. S. J. d. (2009). Differences between intelligent tutor lessons, and the choice to go off-task. In *Proceedings of the 2nd International Conference on Educational Data Mining* (pp. 11-20).

Baker, R. S. J. d., Corbett, A. T. & Aleven, V. (2008). More accurate student modeling through contextual estimation of slip and guess probabilities in Bayesian Knowledge Tracing. In *Proceedings of the 9th International Conference on Intelligent Tutoring Systems* (pp. 406-415).

Baker, R. S. J. d., Corbett, A. T., Gowda, S. M., Wagner, A. Z., MacLaren, B. M., Kauffman, L. R., Mitchell, A. P. & Giguere, S. (2010). Contextual slip and prediction of student performance after use of an intelligent tutor. In *Proceedings of the 18th Annual Conference on User Modeling, Adaptation, and Personalization* (pp. 52-63).

Baker, R. S., Corbett, A. T. & Koedinger, K. R. (2004). Detecting student misuse of intelligent tutoring systems. In *Proceedings of the 7th International Conference on Intelligent Tutoring Systems* (pp. 531-540).

Baker, R. S. J. d., Corbett, A. T., Koedinger, K. R., Evenson, S. E., Roll, I., Wagner, A. Z., Naim, M., Raspat, J., Baker, D. J. & Beck, J. (2006). Adapting to when students game an intelligent tutoring system. In *Proceedings of the 8th International Conference on Intelligent Tutoring Systems* (pp. 392-401).

Baker, R. S., Corbett, A. T., Koedinger, K. R. & Wagner, A. Z. (2004). Off-task behavior in the Cognitive Tutor classroom: When students "game the system". In *Proceedings of ACM CHI 2004: Computer-Human Interaction* (pp. 383-390).

Baker, R. S. J. d., Corbett, A. T., Roll, I. & Koedinger, K. R. (2008). Developing a generalizable detector of when students game the system. *User Modeling and User-Adapted Interaction, 18*(3), 287-314.

Baker, R. S. J. d., Corbett, A. T. & Wagner, A. Z. (2006). Human classification of low-fidelity replays of student actions. In *Proceedings of the Educational Data Mining Workshop at the 8th International Conference on Intelligent Tutoring Systems* (pp. 29-36).

Baker, R. S. J. d. & de Carvalho, A. M. J. A. (2008). Labeling student behavior faster and more precisely with text replays. In *Proceedings of the 1st International Conference on Educational Data Mining* (pp. 38-47).

Baker, R. S. J. d., de Carvalho, A. M. J. A., Raspat, J., Aleven, V., Corbett, A. T. & Koedinger, K. R. (2009). Educational software features that encourage and discourage "gaming the system". In *Proceedings of the 14th International Conference on Artificial Intelligence in Education* (pp. 475-482).

Baker, R. S. J. d. & Gowda, S. M. (2010). An analysis of the differences in the frequency of students' disengagement in urban, rural, and suburban high schools. In *Proceedings of the 3rd International Conference on Educational Data Mining* (pp. 11-20).

Baker, R. S. J. d., Mitrovic, A. & Mathews, M. (2010). Detecting gaming the system in constraint-based tutors. In *Proceedings of the 18th Annual Conference on User Modeling, Adaptation, and Personalization* (pp. 267-278).

Baker, R., Walonoski, J., Heffernan, N., Roll, I., Corbett, A. & Koedinger, K. (2008). Why students engage in "gaming the system" behavior in interactive learning environments. *Journal of Interactive Learning Research, 19*(2), 185-224.

Beal, C. R., Qu, L. & Lee, H. (2006). Classifying learner engagement through integration of multiple data sources. In *Proceedings of the 21st National Conference on Artificial Intelligence* (pp. 2-8).

Beal, C. R., Qu, L. & Lee, H. (2008). Mathematics motivation and achievement as predictors of high school students' guessing and help-seeking with instructional software. *Journal of Computer Assisted Learning, 24*, 507-514.

Beck, J. (2005). Engagement tracing: using response times to model student disengagement. In *Proceedings of the 12th International Conference on Artificial Intelligence in Education* (pp. 88-95).

Buckley, B., Gobert, J. & Horwitz, P. (2006). Using log files to track students' model-based inquiry. In *Proceedings of the Seventh International Conference of the Learning Sciences* (pp. 57-63).

Cetintas, S., Si, L., Xin, Y. P. & Hord, C. (2010). Automatic detection of off-task behaviors in intelligent tutoring systems with machine learning techniques. *IEEE Transactions on Learning Technologies, 3*(3), 228-236.

Clements, M. (1982). Careless errors made by sixth-grade children on written mathematical tasks. *Journal for Research in Mathematics Education, 13*(2), 136-144.

Cocea, M., Hershkovitz, A. & Baker, R. S. J. d. (2009). The impact of off-task and gaming behaviors on learning: Immediate or aggregate? In *Proceedings of the 14th International Conference on Artificial Intelligence in Education* (pp. 507-514).

Corbett, A. (2001). Cognitive Computer Tutors: Solving the Two-Sigma Problem. In M. Bauer, P. J. Gmytrasiewicz & J. Vassileva (Eds.), *Proceedings of the 2001 International Conference on User Modeling* (pp. 137-147). Berlin: Springer.

Corbett, A. T. & Anderson, J. R. (1995). Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Modeling and User-Adapted Interaction, 4,* 253-278.

Frank, E. & Witten, I. H. (1998). Generating accurate rule sets without global optimization. In *Proceedings of the Fifteenth International Conference on Machine Learning* (pp. 144-151).

Fredricks, J. A., Blumenfeld, P. C. & Paris, A. H. (2004). School engagement: Potential of the concept, state of the evidence. *Review of Educational Research, 74*(1), 59-109.

Gong, Y, Beck, J. E. & Heffernan, N. T. (2010). Comparing knowledge tracing and performance factor analysis by using multiple model fitting procedures. In *Proceedings of the 10th International Conference on Intelligent Tutoring Systems* (pp. 35-44).

Hershkovitz, A., Baker, R. S. J. d., Gobert, J. & Wixon, M. (2011). Goal orientation and changes of carelessness over consecutive trials in science inquiry. In *Proceedings of the 4th International Conference on Educational Data Mining* (pp. 315-316).

Hershkovitz, A., Baker, R. S. J. d., Gobert, J., Wixon, M. & Sao Pedro, M. (in press). Discovery with models: A case study on carelessness in computer-based science inquiry. *American Behavioral Scientist*.

Hoerl, A. E. & Kennard, R. W. (1970). Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics, 12*(1), 55-67.

Inventado, P. S. & Numao, M. (2012). Supporting student self-regulation in unsupervised learning environments. In *Proceedings of the 20th International Conference on Computers in Education* (pp. 1-4).

Johns, J. & Woolf, B. (2006). A dynamic mixture model to detect student motivation and proficiency. In *Proceedings of the 21st National Conference on Artificial Intelligence* (pp. 163-168).

Karweit, N. & Slavin, R. E. (1982). Time-on-task: Issues of timing, sampling, and definition. *Journal of Experimental Psychology, 74*(6), 844-851.

Koedinger, K. R., Baker, R. S. J. d., Cunningham, K., Skogsholm, A., Leber, B. & Stamper, J. (2010). A data repository for the EDM community: The PSLC DataShop. In C. Romero, S. Ventura, M. Pechenizkiy & R. S. J. d. Baker (Eds.), *Handbook of Educational Data Mining* (pp. 43-56). Boca Raton, FL: CRC Press.

Lahaderne, H. M. (1968). Attitudinal and intellectual correlates of attention: A study of four sixth-grade classrooms. *Journal of Educational Psychology, 59*(5), 320-324.

Lee, S. W., Kelly, K. E. & Nyre, J. E. (1999). Preliminary report on the relation of students' on-task behavior with completion of school work. *Psychological Reports, 84*, 267-272.

Maris, E. (1995). Psychometric Latent Response Models. *Psychometrika, 60*(4), 523-547.

Martin, J. & VanLehn, K. (1995). Student assessment using Bayesian nets. *International Journal of Human-Computer Studies, 42*, 575-591.

Maydeu-Olivares, A. & D'Zurilla, T. J. (1995). A factor analysis of the social problem-solving inventory using polychoric correlations. *European Journal of Psychological Assessment, 11*(2), 98-107.

Muldner, K., Burleson, W., Van de Sande, B. & VanLehn, K. (2011). An analysis of students' gaming behaviors in an intelligent tutoring system: Predictors and impacts. *User Modeling and User-Adapted Interaction, 21*(1-2), 99-135.

Ocumpaugh, J., Baker, R. S. J. d. & Rodrigo, M. M. T. (2012). *Baker-Rodrigo Observation Method Protocol (BROMP) 1.0. Training Manual version 1.0*. Technical Report. New York, NY: EdLab. Manila, Philippines: Ateneo Laboratory for the Learning Sciences.

Pardos, Z. A., Baker, R. S. J. d., Gowda, S. M. & Heffernan, N. T. (2011). The sum is greater than the parts: Ensembling models of student knowledge in educational software. *SIGKDD Explorations, 13* (2), 37-44.

Pavlik, P. I., Cen, H. & Koedinger, K. R. (2009). Performance factors analysis -- A new alternative to knowledge tracing. In *Proceedings of the 14th International Conference on Artificial Intelligence in Education* (pp. 531-538).

Quinlan, J. R. (1993). *C4.5: Programs for Machine Learning*. San Mateo, CA: Morgan Kaufmann.

Rodrigo, M. M. T., Baker, R. S. J. d., Agapito, J., Nabos, J., Repalam, M. C., Reyes, S. S. & San Pedro, M. C. Z. (2012). The effects of an interactive software agent on student affective dynamics while using an intelligent tutoring system. *IEEE Transactions on Affective Computing, 3* (2), 224-236.

Rodrigo, M. M. T., Baker, R. S. J. d., McLaren, B., Jayme, A. & Dy, T. (2012). Development of a workbench to address the educational data mining bottleneck. In *Proceedings of the 5th International Conference on Educational Data Mining* (pp. 152-155).

Roll, I., Aleven, V., McLaren, B. M. & Koedinger, K. R. (2011). Improving students' help-seeking skills using metacognitive feedback in an intelligent tutoring system. *Learning and Instruction, 21*, 267-280.

Rowe, J., McQuiggan, S., Robison, J. & Lester, J. (2009). Off-task behavior in narrative-centered learning environments. In *Proceedings of the 14th International Conference on Artificial Intelligence and Education* (pp. 99-106).

Sabourin, J., Rowe, J., Mott, B. & Lester, J. (2011). When off-task is on-task: The affective role of off-task behavior in narrative-centered learning environments. In *Proceedings of the 15th International Conference on Artificial Intelligence in Education* (pp. 534-536).

San Pedro, M. O. C., Baker, R. & Rodrigo, M. M. (2011a). Detecting carelessness through contextual estimation of slip probabilities among students using an intelligent tutor for mathematics. In *Proceedings of 15th International Conference on Artificial Intelligence in Education* (pp. 304-311).

San Pedro, M. O. C., Rodrigo, M. M. & Baker, R. S. J. d. (2011b). The relationship between carelessness and affect in a Cognitive Tutor. In *Proceedings of the 4th bi-annual International Conference on Affective Computing and Intelligent Interaction*.

Shih, B., Koedinger, K. R. & Scheines, R. (2008). A response time model for bottom-out hints as worked examples. In *Proceedings of the 1st International Conference on Educational Data Mining* (pp. 117-126).

Shute, V. J. (1995). SMART: Student modeling approach for responsive tutoring. *User Modeling and User-Adapted Interaction, 5* (1), 1-44.

Walonoski, J. A. & Heffernan, N. T. (2006). Prevention of off-task gaming behavior in intelligent tutoring systems. In *Proceedings of the 8th International Conference on Intelligent Tutoring Systems* (pp. 722-724).

Wixon, M., Baker, R. S. J. d., Gobert, J., Ocumpaugh, J. & Bachmann, M. (2012). WTF? Detecting students who are conducting inquiry without thinking fastidiously. In *Proceedings of the 20th International Conference on User Modeling, Adaptation and Personalization* (pp. 286-298).

CHAPTER 15 –Knowledge Component (KC) Approaches to Learner Modeling

Vincent Aleven and Kenneth R. Koedinger
Carnegie Mellon University - Human-Computer Interaction Institute

Introduction

A distinguishing characteristic of ITSs is that they engage in learner modeling, meaning that they estimate key characteristics of each learner based on interactions with the tutoring system. Although learner modeling is not fundamentally different from other forms of formative assessment, some properties of the learner modeling approaches applied in ITSs are not widely shared with other forms of formative assessment. First, learner modeling methods used in ITSs typically assess learners over time and across many measurement moments. Therefore, they tend to take into account that the learner is learning. Second, the assessment is done primarily to support pedagogical decision making by the tutoring system. Third, learner modeling and instruction go hand in hand; the learners are assessed on the same problems on which they are receiving instruction. Finally, although the term "assessment" is typically associated with a focus on learners' knowledge and skills, across the spectrum of ITSs, learner models hold many different types of information.

In this chapter, we focus on models that represent the knowledge targeted in the instruction and also knowledge that may be used to learn target knowledge (e.g., metacognition or motivational beliefs related to learning). These kinds of learner models capture what a given learner knows (tacitly or explicitly) and does not know yet, so that the system can tailor its instruction accordingly. In particular, we focus on approaches that aim to model the learner's knowledge with sufficient specificity that the learner's performance on future tasks can be accurately predicted. In pursuit of this goal, it has turned out to be useful to model the learner's knowledge as a set of inter-related KCs. The Knowledge-Learning-Instruction (KLI) Framework defines a KC as "an acquired unit of cognitive function or structure that can be inferred from performance on a set of related tasks" (Koedinger, Corbett & Perfetti, 2012). Examples of ITSs that have taken a KC approach to learner modeling are Cognitive Tutors (Anderson, Corbett, Koedinger & Pelletier, 1995; Koedinger & Aleven, 2007), constraint-based tutors (Mitrovic, Mayo, Suraweera & Martin, 2001), and Andes (VanLehn et al., 2005).

The KC approach to learner modeling comes with a number of key challenges. First, in any domain of interest, how can we determine what the KCs are that learners acquire? Put differently, how can the targeted expertise be decomposed into a set of KCs that have psychological reality, as evidenced by the fact that the model can be used to accurately predict student performance across tasks and over time? Second, given a KC model, how can a tutoring system accurately determine, for any learner at any point in time, what that learner's level of mastery is for each of the individual KCs? Third, given an accurate model of an individual learner, how can an ITS tailor its instruction so learners learn more effectively and/or efficiently? In this chapter, we focus primarily on the first question while briefly touching on the other two. A key point in our chapter is that automated approaches to creating and refining KC models can substantially enhance "manual" approaches.

In the KC approach to learner modeling, building an intelligent tutor for a new domain requires a KC model. Since existing KC models are rarely available or "just right," typically, a new model needs to be built. Doing so, however, requires a substantial amount of effort. In a traditional approach to KC modeling, an author creates a model by hand, based on careful cognitive task analysis. Ideally, the author would conduct both theoretical task analysis and empirical task analysis (Lovett, 1998). In theoretical cognitive task analysis, an author elucidates the knowledge demands of a task by carefully analyzing its structure (e.g., its possible goal/subgoal hierarchies). In empirical task analysis, by contrast, an author collects data about problem solving in the given task domain, employing methods such as think-alouds, interviews of experts, analysis of errors novices make on written tests, difficulty factors analysis (DFA), and so forth (Baker, Corbett & Koedinger, 2007).

Given that creating a KC model tends to be labor intensive, ITS authoring tools support aspects of cognitive task analysis and KC modeling. In this chapter, we review how one set of authoring tools (the Cognitive Tutor Authoring Tools or CTAT, for short) support theoretical cognitive task analysis and KC modeling as an integral part of the process of building a tutor, with substantial gains in authoring efficiency (Aleven, McLaren, Sewall & Koedinger, 2009). However, it is becoming clear that manual KC modeling does not always yield optimal KC models. Manual KC modeling efforts have produced multiple cases where KC models created or refined through automated or semi-automated methods did better (e.g., on predicting student performance) than real-world models created by hand. For example, our original assumption in building the cognitive model (a kind of KC model) for story problem solving within Cognitive Tutor Algebra was that story problems are hard because it is challenging to comprehend a story and translate it to an equation. Without collecting data, these hand-crafted models would have been wrong. For beginning algebra students, introductory story problems are not solved using equations and are actually easier to solve than the matched equation (Koedinger & Nathan, 2004). Further, the challenge in translating a story to an equation is more in the difficulties in producing the symbolic equation than in comprehending the English (Koedinger & McLaughlin, 2010). As a second example, in the Geometry Cognitive Tutor, the cognitive model in the fielded tutor did not make a distinction for “decomposition problems” between those where the decomposition was scaffolded and those where it was not. Data mining showed a large and important difference between these two situations, however (Stamper & Koedinger, 2011), and a tutor redesigned based on the data-discovered new model yields more efficient and effective learning (Koedinger, Stamper, McLauglin, Nixon, in press). Third, across 11 datasets, an automated model refinement algorithm called Learning Factors Analysis (LFA) found better cognitive models than the original models and than best data-driven models (Koedinger, McLaughlin & Stamper, 2012).

Part of the problem may be that in practice, the manual approach to KC modeling relies too much on theoretical task analysis and not enough on empirical task analysis. It may be fair to say that the first cut at a KC model is often created without much empirical task analysis. In the process of developing a tutor, it can be difficult to find the time to collect data with students from the target population. A second reason why hand-crafted models can be suboptimal is that even common methods for empirical cognitive task analysis may not be able to address all modeling decisions that arise (often, there are simply too many of them) and may not uncover subtle over-generalizations and under-generalizations that novices make while learning new material. Such discoveries may require larger data sets than those typically collected in empirical task analysis, such as those collected as tutor log data.

As second focus of the chapter, therefore, is on methods and tools for discovering or refining KC models using data on task performance (e.g., tutor log data). Specifically, we concentrate on three data-driven and/or machine learning approaches: (1) use of visualization and statistical modeling tools for analysis and refinement of existing models based student log data for example, by using the DataShop (Koedinger et al., 2010); (2) combining human expertise, machine learning, and log data to automatically discover better KC models (LFA) (Koedinger et al., 2012), again using tutor log data, and (3) using machine-learning-based model of student learning (e.g., SimStudent) to learn an accurate KC model from being tutored in the domain (Li, Matsuda, Cohen & Koedinger, 2011).

Background: What Is KC Modeling?

In this section, we review the notion of a KC as a central element in learning modeling. In the next section, we review some of the key evidence in the ITS literature in support this approach – instances in which a KC-based learner model was used in an ITS to adapt instruction with measurable impact on student learning.

But first, what is a knowledge component? Does any decomposition of the knowledge needed for a particular task result in a KC model? And what does it mean for a KC model to have psychological reality? As mentioned, in the KLI framework (Koedinger et al., 2012), a KC is an acquired unit of cognitive functioning, which (being a latent construct) has to be inferred from learners' performance. This definition encompasses a wide range of knowledge types, not just procedural knowledge. A complex cognitive task is viewed as involving many fine-grained KCs (Anderson, 1993). These KCs are learned separately in the sense that practice with one does not cause improved mastery of another. While there is no possibility of *transfer from one KC to another*, KCs provide a good way of thinking about *transfer between problems or problem steps.* In many domains (e.g., mathematics) different problems require use of overlapping sets of KCs. One expects to see transfer between different problems or problem steps, in the sense that practice on one leads to improved performance on the other, exactly to the extent that there is overlap in the KCs needed for these problems or steps. A key characteristic of the KC approach to learner modeling (which is not shared with other approaches to assessment, even prominent ones such as IRT analysis, cf., Wilson & De Boeck, 2004) is that such transfer relations are explicitly accounted for. The KCs are considered to be the units of transfer.

We can now state what it means for a KC model to have psychological reality. Put simply, it means the model can be used to make accurate predictions of a given student's performance on future problems based on that student's performance on past problems. Put differently, it means that the transfer predictions that are implied by the model are actually observed in data about student performance, typically, tutor log data (Aleven, 2010). This description leaves open how a KC model can be used to predict future performance. There are multiple ways of doing so; below we describe one often-used method. A key pre-requisite for making accurate predictions is that the model captures knowledge at the right level of generality.

It may be helpful to contrast the KC approach to learner modeling with an alternative but incorrect viewpoint, namely, that "students learn what they spend time on." While that statement may have a ring of truth to it, it is actually misleading. It depends critically on how we categorize student activities. When we look at student activity at the level of problems, we may go wrong. We need to categorize problem-solving steps, not problems, and we need to do so with respect to their cognitive demands. In a study on a tutor unit with geometry area composition problems, model discovery revealed a different skill is needed for unscaffolded decomposition steps than for scaffolded ones (Stamper & Koedinger, 2011). In these decomposition steps, students must realize that they can compute the area of irregular geometric shapes as the sum or difference of the area of two different regular shapes. In the fully-scaffolded tutor, students did not need to plan to decompose the irregular area shape on their own, as the tutor prompted for the areas of the regular shapes and then for their sum or difference. A redesigned tutor included unscaffolded problems where students were not prompted to decompose the figure; they basically had to figure it out themselves, as one big step in the tutor problem. New problems that isolate the decomposition planning step and did not require its execution were also created and used in the redesigned tutor. The students in the redesigned tutor condition learned better (Koedinger, Stamper, McLaughlin & Nixon, in press). All of the problems that both groups solved in this study were composition problems, and the students who worked with the scaffolded tutor spent more time on these problems overall. Thus, by the simple statement above, they should have learned the decomposition skills better. They did not. Students using the redesigned tutor learned the decomposition skill better, because only they had practiced that skill – in the scaffolded group, the tutor's scaffolding took over a critical part of the work. Generally, students learn the KCs they spend time practicing. However, what these KCs are is not obvious. KCs are not directly observable and most are not open to conscious reflection, despite our strong feelings of self-awareness of our own cognition. They can, however, be inferred and discovered from student performance data across multiple tasks via a statistical comparison of alternative categorizations, that is, of alternative KC models.

As another contrasting case, we consider the ALEKS system and its underlying knowledge space algorithm (Falmagne, Koppen, Villano, Doignon & Johannesen, 1990). ALEKS is a commercial system for mathematics practice, focused on individualized problem selection to diagnose a student's knowledge state. It presents problems to students and provides mathematical tools to solve them (e.g., digital version of a compass, on-screen calculator, grapher, etc.) but does not provide the step-by-step guidance (or equivalently, an "inner loop") characteristic of ITSs (VanLehn, 2006, 2011).

The adaptive problem selection technique in ALEKS relies on a "knowledge space" created offline that captures presumed precedence relations among problem types. That problem type A precedes (i.e., is a prerequisite of) type B is determined empirically. In essence, A is assumed to precede B if most students that get B right also get A right and if of those that get B wrong many get A right. In other words, evidence that a student knows B implies they know A. These precedence relations define a partial order among all problem types and this partial ordering is used to diagnose a student's knowledge state, defined as the set of problem types that the student masters (i.e., can solve correctly). From the knowledge state, ALEKS can select an appropriate next problem – one of the problem types that follow those on the edge of what that student knows.

The KC approach described in the current chapter and ALEKS' approach using knowledge spaces share an important goal, namely, learner modeling in support of individualized problem selection. They also share the view that student performance data is a valuable source of information for model building beyond intuition or analysis. A key difference is that the KC approach attempts to explicate what the KCs are, whereas there is no such attempt in the knowledge space approach. Whereas the ALEKS approach uses data *in place* of theory and intuition, the KC approach uses data *in addition to* theory and intuition. To illustrate the difference, the knowledge space algorithm will identify A as a prerequisite of B if A is consistently easier than B (for the same students)[9]. However, unless the KCs used in A are also needed in B (e.g., A is finding factors of a number and B is finding the least common multiple of two numbers, which requires finding factors), the statistical regularity used to infer a prerequisite may be spurious. A second reason why the KC model approach values explication of the KCs is that the resulting explanatory model may not only enhance generalization, but can better be interpreted for use in course redesign (Lovett, Meyer & Thille, 2008) and in tutor redesign (Koedinger & McLaughlin, 2010; Koedinger, Stamper, McLaughlin & Nixon, in press) .

It is worth noting that a knowledge space approach is not incompatible with a KC modeling approach and, indeed, we would recommend analysts using a knowledge space to try to explicate the in-common KCs implied by the links in the prerequisite graph. Links should be eliminated (or not produced) when a plausible KC cannot be generated.

[9] Note that the knowledge space algorithm is more nuanced then this treatment might imply: It is possible for A to be easier than B, but still have a number of students getting B right and A wrong and if that happens enough, a precedent relation from A to B will not be inferred.

Use of KC-Based Learner Models and Evidence of Effectiveness

To motivate the KC approach, we briefly touch on the second and third key challenges identified in the introduction: Once a KC model has been created for a given task domain, how can an ITS assess an individual student's mastery of the KCs in the model? Also, what are good ways of using that assessment to individualize instruction? We review some of the key studies in the literature.

A widely used approach to tracking individual learners' knowledge growth using a KC-based model is BKT (Corbett & Anderson, 1995). In this approach, the posterior probability of mastering a KC is updated each time a student encounters a problem step involving that KC. The posterior probability depends on the performance on that step, the prior probability of mastery, the likelihood of learning from a step, and conditional probabilities that allow for the possibility that the student may guess the step or slip. These parameters are typically set separately for each KC and are sometimes estimated from data (Corbett & Anderson, 1995). As an alternative to BKT, Bayesian Networks are often used to maintain KC-based learner models (Conati, Gertner & Vanlehn, 2002). Further, recent EDM research has developed new methods for tracking individual learners' knowledge growth in terms of KCs (e.g., Gong, Beck & Heffernan, 2011; Lee & Brunskill, 2012). Some of these methods have been shown to have greater predictive accuracy than BKT. To the best of our knowledge, these methods have not yet been used for online learner modeling or online individualization of instruction, although one of them, AFM (described below) has been used offline for KC model and tutor improvement, and there is much ongoing work in this area.

The primary use of KC-based learner models in ITSs is to support individualized problem selection (Corbett, McLaughlin & Scarpinatto, 2000; Corbett & Anderson, 1995; Mitrovic & Martin, 2004). In this approach, the system uses its learner model to select problems that (for the given student, at the given point in time) target unmastered KCs. It continues to do so until the learner model indicates that the learner has reached mastery of the targeted KCs. This approach to individualized problem selection, often called "Cognitive Mastery," is used in Cognitive Tutors, a type of ITS used widely in American mathematics education (Anderson et al., 1995; Koedinger & Aleven, 2007). Individualized problem selection based on a KC model can be very effective in enhancing student learning. In studies with the Lisp Tutor, an early Cognitive Tutor, Corbett et al. (2000) found substantial improvement in student learning outcomes, due to this form of individualization, with on average only a very modest increase in the time spent. Also, a study by Cen et al. (2007) showed that improving the accuracy of learner modeling – by tuning the parameters of the BKT procedure – leads to more efficient learning by students. These studies on individualized problem selection provide key evidence that adaptive individualization by an ITS enhances student learning, an important argument in favor of this advanced learning technology.

KC-based learner models have also been used for individualizing a number of other aspects of tutoring systems. For example, they have been used to individualize ways in which worked examples are used in ITSs. In the SE-COACH by Conati and VanLehn, a KC-based learner model is used to decide what steps in a worked example a particular learner should be prompted to explain, which for early learners was shown to be helpful, compared to a system in which the steps to explain were not selected on an individual basis (Conati & Vanlehn, 2000). In other work, a KC-based learner model was used to select, on an individual basis, suitable examples for analogical comparison (Muldner & Conati, 2007). In a study by Salden and colleagues, a KC-based learner model was used to transition adaptively (KC by KC) from studying worked examples to solving problems. In a lab study and a classroom study, this adaptive technique led to better or more efficient learning (Salden, Aleven, Renkl & Schwonke, 2009). KC-based learner models have also been used as components in models of metacognition, affect, and specific (desirable or undesirable) learning behaviors. For example, in work on modeling adaptive help seeking behavior (a key metacognitive skill), the learner's decision to seek help depends on the student's level of mastery of the given KCs. Help requests are deemed to be adaptive only on steps that involve KCs that are unfamiliar to the student at the given point in time (Aleven, Roll & Koedinger, 2012; Roll, Aleven, McLaren & Koedinger, 2011). Further, in work on creating machine-learned models of various aspects of learners and learner behaviors (e.g., gaming the system, off-task behavior, affective states, etc.) the level of mastery of a KC is one of the features that serves as input to the machine-learned classifier (Baker et al., 2006; Baker, Goldstein & Heffernan, 2011). Finally, KC models are often used as open learner models (OLMs) (Bull & Kay, 2007; Long & Aleven, 2013; Mitrovic & Martin, 2007), which have become a common feature of ITS. The OLM communicates a sense of progress and may support useful self-assessment and reflection. A recent study showed higher post-test scores in a tutor for linear equation solving with an OLM, compared to a version without the OLM (Long & Aleven, 2013).

One way to generalize from this body of empirical work is to say that individualization of instruction based on a KC-based learner model has been shown to be especially effective in the system's outer loop (VanLehn, 2006), as studies on individualized problem selection and example fading indicate, while also being somewhat effective (but not as effective) in the inner loop. This generalization is quite coarse, however – future work may provide a more nuanced perspective.

Tutor Authoring Tools and the KC Approach to Learner Modeling

Let us now return to the first of the three challenges listed in the introduction of the chapter: How can accurate KC models be created? In the remainder of the chapter, we discuss the pros and cons of a variety of approaches, starting with ITS authoring tools that support a "manual" approach to KC modeling. By manual we mean an approach unaided by data mining or machine learning algorithms, although it could involve data collected in the course of doing cognitive task analysis. There are many ways in which authoring tools for ITS can support a manual KC approach to learner modeling. We see two broad areas of functionality. First, tools can provide support for defining/building KC models as part of an iterative tutor development process. This kind of tool support is used offline, when authoring or refining a tutor. Second, authoring tools can provide mechanisms that tutors can use at run time to maintain/update KC-based learner models and tailor aspects of the instruction. While there are many ITS authoring tools (Murray, Blessing & Ainsworth, 2003), we focus on how these two areas of functionality are implemented in CTAT, a tool suite with which they have been involved in the past 10 years (Aleven et al., 2009). Many of our points hold for other ITS authoring tools as well.

CTAT is suite of tools for ITS authoring that has been used to build a wide range of tutors, many of which have been used in actual classrooms or other real educational settings. These tools support KC modeling as an integral part of tutor development. CTAT supports the authoring of two types of tutors: example-tracing tutors and Cognitive Tutors, both of which reflect a KC approach to instruction and learner modeling. Example-tracing tutor rely on behavior graphs to evaluate and interpret student behavior, cognitive tutors rely on a rule-based cognitive model. We view both behavior graphs and rule-based cognitive models as a form of KC models.

A behavior graph is a map of the solution space for a given problem (Koedinger, Aleven, Heffernan, McLaren & Hockenberry, 2004; Newell & Simon, 1972). Creating behavior graphs is a key activity in creating an example-tracing tutor. The tutor uses the graph to interpret student behavior (Aleven et al., 2009). As a first step toward creating such a KC model, an author can use a CTAT tool called the Behavior Recorder to record a behavior graph for a given problem, simply by demonstrating the solution paths on the tutor interface designed for the given problem type. The Behavior Recorder records the demonstrated steps in a graph; the graph may have multiple solution paths corresponding to different ways of solving the problem. The links in the graph represent the steps, and the nodes represent problem-solving states. Behavior graphs are not in and of themselves KC models, however, until the author annotates the links with KC labels. These labels (essentially, names of hypothesized KCs) characterize the knowledge each step is hypothesized to require. Thus, the Behavior Recorder helps an author in coming up with a decomposition of the knowledge in the given task domain, as a form of rational task analysis. A key assumption is that – in creating a KC model – it helps to think at the grain size of problem steps rather than whole problems and it helps to have laid out (and thought about) these steps explicitly. Creating a mapping between steps and KCs can be viewed as implicitly formulating transfer predictions (Koedinger et al., 2004). Annotating two steps with the same KCs implies that one expects to see full transfer between these steps; practice on one improves the other, and vice versa. Conversely, assigning different KC labels to two steps implies that one does not believe such transfer will occur. Partial transfer between steps can be modeled by annotating steps with multiple KCs, some of which are shared between the two steps.

Although behavior graphs labeled with KC names are useful KC models, one limitation is that they do not explicitly state the conditions under which the KCs apply. Put differently, the KCs are defined intensionally, by labeling steps in solution graphs, but not extensionally, by stating applicability conditions. This state of affairs is not ideal, because it means that the exact range of transfer for any given KC is not defined. Rule-based cognitive models (the second kind of KC model an author can create with CTAT) address this limitation (Aleven, 2010). Rule-based cognitive models are computer-runnable simulations of student thinking that can solve tutor problems in the ways that students do. In these models, KCs are expressed in IF-THEN format. The IF-part captures the conditions under which a KC is applicable – in other words, it captures how far the KC transfers. Thus, in creating a rule-based model, an author is forced to think hard about these conditions, more so than when creating example-tracing tutors. CTAT provides a host of tools that help in creating production rule models, including tools for visualizing and debugging production rule models (Aleven, McLaren, Sewall & Koedinger, 2006). Interestingly, in the process of creating a rule-based model, it is often very useful to have behavior graphs with KC labels. These graphs are a specification of how the production rule model should behave. CTAT also supports the use of the behavior graph for testing of rule-based cognitive models.

In addition to supporting the development of KC models, CTAT supports the use of such models within running tutors for purposes of assessment, learner modeling, and individualization. Both example-tracing tutors and Cognitive Tutors built with CTAT use their KC model in the inner loop to assess student problem solving and interpret it in terms of KCs. Both types of tutors are capable of tracking learners' knowledge (i.e., their mastery of the KCs in the tutor's KC model) over time using BKT, briefly described above (Corbett & Anderson, 1995). They also provide individualized problem selection based on Corbett's cognitive mastery mechanism, also described above (Corbett et al., 2000). Thus, as one would expect from an ITS authoring tool, CTAT supports use of the learner model for outer loop decision making, namely, problem selection (VanLehn, 2006). It does not support use of the learner model in the inner loop to individualize instruction, given the limited empirical evidence that inner loop adaptivity leads to improved student learning. Nonetheless, it will be helpful if ITS authoring tools can support use of a learner model to individualize various aspect of the inner loop (e.g., deciding what type of hints to give (cf. Arroyo Beck, Woolf, Beal & Schultz, 2000), if only so that more studies can be done to evaluate what kinds of inner loop individualization are most effective.

Returning to an issue raised in the introduction, the CTAT tools for KC modeling primarily support theoretical cognitive task analysis. Behavior graphs and production rule models help in thinking about what knowledge may be needed to solve particular problems, how that knowledge might be decomposed to capture distinctions that students might make, and how widely specific KCs will transfer. It is left up to the author however whether or not to do so with the aid of empirical task analysis. The tools themselves do not directly support empirical task analysis or the use of data to help create KC models, with one exception: a tool called "novice bootstrapping" supports a process in which behavior graphs are created directly from log data of novice problem solving, rather than from expert demonstrations (Harrer, McLaren, Walker, Bollen & Sewall, 2006). This capability is though to be useful especially for problems with large solution spaces or in ill-defined domains. While CTAT proper provides limited support for using data in tutor or KC model development, it is fully integrated with the DataShop (Koedinger et al., 2010), which provides many tools for this purpose.

Tools for Data-Driven KC Model Validation and Refinement

Once a tutor has been created, log data from the tutor can be used in a variety of data-driven methods to evaluate how well the tutor's KC model captures the psychological reality of student problem solving in the given task domain and how it might be refined/improved. We look at a set of tools we have developed called the DataShop. The DataShop is geared toward supporting a KC-based approach to learner modeling. The DataShop is a large, open, online repository for process data from ITSs and other advanced learning technologies. In addition to being a repository with data sets available for secondary analysis, the DataShop supports KC model analysis, discovery, refinement, and testing. It also offers a standardized format for tutor log data. Given that the DataShop offers an extensive set of tools for KC modeling, logging in DataShop format is a useful feature for any ITS authoring tool. CTAT is fully compatible with the DataShop in this sense. That is, all tutors built with CTAT write detailed logs of student-tutor interactions in DataShop format, without extra work by the author.

Using DataShop tools, an author can perform the following:

- Visually inspect learning curves for individual KCs or groups of KCs extracted from log data. Analysis of a KC model often starts with visual inspection of the learning curves. The learning curves may be based on an existing KC model, such as the one that the given tutor is using, or a new model created by hand. Learning curves capture the performance of a group of students on successive opportunities to apply the given KC or KCs. The learning curves show how challenging the KC is for the target population and the rate at which it is learned. Learning curves are useful tools for visualizing student learning with an ITS, both for researchers and developers. The visual appearance of the learning curves can indicate whether the KC model is psychologically real, with smooth, gradually declining curves suggesting that it is, and ragged or flat curves suggesting it might not be.

- Evaluate or validate a KC model by testing how well it predicts performance in the given log data set. As described below, the DataShop uses a logistic regression model called AFM to predict student performance based on a given KC model (Cen et al., 2006; Spada & McGaw, 1985).

- Search for explanations and newly hypothesized model features when learning curves fail to have their theoretically predicted shape (e.g., are ragged) or when a KC model has poor fit with the data (i.e., leads to inaccurate performance predictions using AFM). An author may try to pinpoint where in the tutor's problem set deviations from ideal learning curves occur, so as to come up with hypotheses for how the KC model might be improved. For example, an author might look at problem steps that are notably easier or notably more difficult than the theoretically predicted value. This type of analysis is supported by a DataShop tool called the Performance Profiler.

- Compare learning curves or learning rates (obtained from fitting AFM) across conditions in an experiment, e.g., does one condition lead to a more efficient or more effective learning *process* than another? (cf., Mathan & Koedinger, 2005).

- Compare how well different KC models fit the same data set (using the AFM). This type of analysis is a way of exploring transfer or lack thereof or equivalently, of exploring the level of generality/specificity at which students learn skills or acquire knowledge. Comparing alternative KC models may help an author understand student learning and may be a key step towards finding a better KC model. An author may create alternative models by hand and upload them into the DataShop. The DataShop presents a "score board" ranking the different KC models for a given data set in terms of their predictive accuracy.

- Analyze what errors are frequent in the data set, for purposes of both tutor improvement (e.g., an author might add error feedback messages for the most common or the most confusing errors) as well as model exploration focused on errors.

- Automatically refine a KC model by running an automated best-first search procedure over a space of model variations; this procedure, called LFA, is described below.

Now let us turn to the question how KC models can be evaluated beyond visual inspection of learning curves, a fundamental concern in the KC modeling approach. Above we said that KC models should be judged by their ability to support accurate predictions of student learning across tasks and over time, but we left open the question how exactly a model can be used to make predictions. Although there is not a single right way of doing so, a good way is to use a logistic regression model known as the AFM, shown in Figure 15-1 (Cen et al., 2006; Spada & McGaw, 1985). This model is used in the DataShop.

$$log \frac{p_{ij}}{1-p_{ij}} = \theta_i + \sum_k \beta_k Q_{kj} + \sum_k Q_{kj}\, \gamma_k\, T_{ik}$$

Given:

p_{ij}	(0 or 1) probability that student *i* gets step *j* correct
Q_{kj}	(0 or 1) whether KC *k* is needed for step *j*
T_{ik}	number of opportunities student *i* has had to practice KC *j*

Estimated:

θ_i	proficiency of student *i*
β_k	ease of KC *j*
γ_k	gain for each opportunity to practice KC *j*

Figure 15-1. AFM built into the DataShop for using a KC model to make predictions about student performance

AFM can be used to estimate the probability that the student will get a step in a tutor problem right, based on that and other students' history on problem steps that involve the same KCs. Each step is assumed to involve one or more KCs. Thus, to use the model it is necessary to have a mapping between tutor problem steps and KCs. In a running ITS, this mapping is typically created in the system's inner loop (VanLehn, 2006), using the tutor's KC model; the mapping is then recorded in the tutor log data. In offline approaches to KC model validation, a static "Q matrix" is often used, which specifies the mapping between steps and KCs. This mapping is essentially what is meant by a KC model in this context. A mapping created dynamically by an ITS can be more flexible than a Q matrix, in the sense that the KCs in a step can depend on the particular solution path the student takes through the problem (Aleven, 2010).

AFM estimates the log odds of the probability that the student will get the step right as the sum of three key quantities: the proficiency of the student (i.e., how good the student is in general, across all KCs, or θ_i), the ease of each KC involved in the step (β_k), and how much the student has learned on prior opportunities to practice each of these KCs ($\gamma_k T_{ik}$). Thus, the model captures effects of practice over time and takes into account that a student is learning, essential characteristics of learner modeling approaches in ITS. Further, the model assumes that practice with a given KC does not have any direct effect on other KCs, a fundamental assumption in KC modeling discussed above. This regression model also makes a number of simplifying assumptions. First, it assumes that all students learn the same KCs. Second, it assumes that a KC that is more difficult than other KCs is so for all students. That is, the model does not allow for the possibility that a particular KC is harder than other KCs for one student, but easier than other KCs for other students (i.e., the model has no student × KC interaction term). Also, the model assumes that the effect of past practice opportunities is measured accurately solely by the number of such opportunities, and that there is no additional information to be gained for example from how successful these past opportunities were. In spite of these assumptions, the model has the virtue of (relative) simplicity and works well in practice. Addressing some of these assumptions leads to more complex models, as has been explored in later work by various researchers (e.g., Gong, Beck & Heffernan, 2011).

In order to use the model to predict student performance, it is necessary to estimate the values for the parameters in the model, specifically, the proficiency of each student (θ_i), the ease of each KC (β_k), and the learning rate for each KC (γ_k), as shown in Figure 15-1. These parameters can be estimated through standard logistic regression modeling techniques, such as generalized linear modeling (in R, the glm function with family= binomial). With such parameter estimates in hand, it is possible to calculate the predictive accuracy of the model on held-out data (i.e., cross-validation). Alternatively, predictive fit indices such as Akaike information criterion (AIC) and Bayesian information criterion (BIC) can be used, especially when the focus is on comparing alternative KC models for the same data set.

How does a KC model's psychological reality relate to its predictive ability? The short answer is, if a KC model is incorrect in the sense that it does not capture the true KCs that students acquire, then the parameter estimates obtained through logistic regression process described above will be inaccurate and therefore lead to inaccurate predictions. To illustrate, consider a model that models KCs at too high a level of abstraction, meaning that it misses distinctions that students actually make. This model models as one KC what for students are two separate KCs. To make the example concrete, consider a model for geometry problem solving that uses one KC for steps that use the circle area formula. Let us assume, however, that in reality, using the circle area formula when the radius is given is different for students from using it when the area of the circle is given – going from area to radius involves a square root, a difficult mathematical operation. As a consequence, the β_k (ease of KC) estimate for the overly abstract KC in our model would be a rough average of the β_k's of the two true KCs, which may have different true β_k's. Similarly, the learning rate parameter (γ_k) for the overly abstract KC might also not accurately reflect those for the true KCs. Similar comments can be made for a model with overly specific KCs. In a model with overly specific KCs (i.e., a model that misses abstractions that students actually make), the practice opportunities are not assigned correctly to the true KCs, which will lead to inaccurate parameter estimates.

Automated Methods for KC Modeling Learning Factors Analysis (LFA)

While tools for data-driven KC model analysis and validation such as those provided by the DataShop are very helpful, a model author still needs to do a substantial amount of work to come up with model variations and test whether they have greater predictive accuracy than the model from which they were derived. In any realistic KC modeling effort, there may be many modeling decisions about which there is uncertainty. As a result, the space of models that might potentially do a better job than an initial model created by hand is often quite large. It would be difficult for an author to optimize a model by searching this large space by going through the manual (though tool-supported) model refinement process described in the previous section. Therefore, EDM research has started to produce automated methods for discovering and refining KC models.

LFA is one such method (Cen et al., 2006). It automates the process of hypothesizing alternative cognitive models and testing them against data. This method searches a space of KC models, given (1) an initial model, (2) information from which model variations can be generated, and (3) a data set with step-level problem-solving data from a group of students on relevant problems (a typical tutor log data set, in other words, such as those captured in the DataShop). The method does a best-first search, using AFM as described above to evaluate each model variation and using the AIC metric for relative model fit.

To use LFA for model refinement, an author must provide information that the LFA algorithm uses to derive new KCs, item b in the list above. To this end, an author needs to identify factors in the given problem set that are hypothesized to make a difference in how students view or approach problems and therefore in the KCs that they acquire when they work through these problems. For each of these hypothesized difficulty factors, the author also needs to specify in which problems that factor occurs, a so-called "P matrix." The LFA algorithm uses this information to split KCs into more specific KCs, based on the difficulty factors, resulting in a finer-grained model. The algorithm also is capable of merging two KCs under appropriate circumstances, resulting in a coarser-grained model. The new model (with the split and/or merged KCs) is evaluated using the AFM described above and it is retained in the best-first search process if the AIC criterion shows improved fit over the model from which it was derived. The LFA algorithm can be run, on request, on any DataShop dataset and has been run on tens of datasets.

As a hypothetical example, a KC model for geometry problem solving might have a KC for applying the isosceles triangle theorem to infer the measure of one of the two angles opposite the congruent sides, given the measure of the other such angle. (The two angles are congruent.) A model author might wonder whether it makes a difference whether the isosceles triangle is acute or obtuse – students in the target population may be more familiar with acute triangles and recognize them more readily as being isosceles. Similarly, an author might wonder whether the triangle's orientation in the diagram matters – the third, non-congruent side may be at the bottom (fir tree configuration) or at the top (ice cream cone) or one of the congruent sides may be at the bottom (lying on the side). Perhaps the fir tree configuration is more familiar and recognizable for students. Our author would therefore define two difficulty factors, one capturing the obtuse/acute distinction, the other the tree/cone/on-the-side distinction. These two factors together would yield 12 possible ways of refining the KC model for the isosceles triangle skill, all more fine-grained than the original model. The LFA search would test if any of the more fine-grained KC models better accounts for the student data that was observed, based on its AIC score. The example illustrates how the space of models can be quite large, which is a key reason why automated methods are useful.

In a study by Koedinger, McLaughlin, and Stamper (2012), the LFA method was applied to 11 DataShop data sets from a variety of Cognitive Tutors and an educational game in a variety of domains. Models generated by LFA were found to improve upon handmade models in all 11 cases, demonstrating the potential of the method for automated model generation. Among the improved models was one for problem solving with the geometry area formula. As described above, the method discovered that for circle area problems, it mattered whether the radius of the circle or the area of the circle was given, whereas for other formulas, it did not make a difference whether the formula was applied forwards or backwards – only the circle formula required use of the square root when applied backwards.

A limitation of LFA is that it needs an initial KC model and therefore does not obviate the need for manual KC modeling. Some attempt at indicating categories of problem (steps), as simple as topics or learning objectives, is necessary. More desirable is to use LFA as supplement to cognitive task analysis done in the early stages of tutor building. Further, LFA cannot compensate for limitations in the breadth of the problem-solving tasks used in a tutor. Task difficulty factors that have a large effect on student performance can be recognized with just one or a few problems involving them. For LFA to detect smaller effects, however, a larger number of problems with and without a difficulty factor are needed. This frequency and variety may or may not occur when a problem set has not been designed with these difficulty factors in mind. Finally, better results can be achieved if there is at least some randomness in the order in which problems are assigned to students, which is not always the case in tutor problem sets.

Use of Mixed-Initiative Machine Learning for KC Modeling

As a final tool for KC modeling, we look at a machine-learning approach to rule-based KC modeling. As mentioned, advantages of rule-based models over other types of KC models are that transfer predictions are made explicit in the conditions of the rules (i.e., in their IF part, given they are written in IF-THEN format) and that they deal well with problems that have a large solution space, as occurs for example in algebraic problem solving (Waalkens, Aleven & Taatgen, 2013). However, writing rule-based KC models requires a substantial amount of work and technical expertise (i.e., AI programming skill). The machine learning approach described in this section helps address this shortcoming.

Li, Matsuda, Cohen, and Koedinger (2011) developed an approach that automatically discovers student models using a state-of-art machine-learning agent, SimStudent. They show that the discovered model is of higher quality than human-generated models and demonstrate how the discovered model can be used to improve a tutoring system's instruction strategy. SimStudent learns a rule-based KC model, of the kind that CTAT authors can create by hand, by being tutored. An author (e.g., a subject matter expert without programming expertise) interacts with SimStudent by first presenting it with a problem using a CTAT interface (e.g., an equation to solve like 100–4x=40). SimStudent uses any productions it has learned so far to try to generate a first step in solving the problem. If none apply to the current state, SimStudent asks the author to demonstrate the next step. It then uses multiple inductive learning mechanisms to learn the IF-part (information retrieval and conditions of applicability) and THEN-part (a function sequence) of a production rule that can reproduce the demonstrated step and generalize to other similar steps. If SimStudent has a production that applies (its IF-part matches the current state), it displays the resulting action (by executing the THEN-part) in the CTAT interface to be checked by the author. If the author confirms the step, this action is added as a positive example of the production. If the author rejects the step, the action is added as a negative example and the productions IF-part will be refined (e.g., by adding a condition that stops the rule from firing in states like this one in the future). SimStudent tries again, perhaps grounding out in asking for a demonstration of the next step.

The author may (but need not) provide guidance to SimStudent about what steps to try to generalize across by providing the same skill label to these steps, similar to labeling steps in behavior graphs. Even when a label is provided for a set of steps, SimStudent may not find a single general production that works across all same-labeled steps. In those cases it creates multiple productions and these, then, are the key basis for theoretical discovery of a better cognitive model (these are an automated generation of the equivalent of a factor used to perform a split in LFA). For example, in Li et al. (2011), equation-solving steps involving dividing by a coefficient (e.g., 3x=12, 15=–5x, or –x=8) were all given the same label (meaning that the author hypothesized that these steps involve the same KC). However, SimStudent learned separate productions for steps where there is number before the x (e.g., 3x=12 or 15=–5x) vs. steps where there is just a "–" sign before the x (e.g., –x=8 or –13=–x). In other words, SimStudent, at its own initiative, refined the KC model proposed by the author, by splitting the author's hypothesized KC. The KC model created by SimStudent (it made other novel distinctions besides this one) was compared (using AFM) with the best existing hand-created model. It led to better prediction fit as measured both by AIC and root-mean-square error (RMSE) on the test set in cross-validation. In particular, the cognitive model SimStudent learned better captured that human students find the second category of problems, with no explicit numerical coefficient (e.g., –x=8 or –13=–x), to be much more difficult than the first category (60% vs. 30% error rate). This kind of cognitive model discovery via SimStudent has now been tested across multiple domains: algebra equation solving, fraction addition, chemistry stoichiometry problems, and English article choices.

In some cases (e.g., algebra equation solving), the SimStudent-discovered cognitive model beat the prior best hand model and the prior best LFA model. However, adding the SimStudent discovered factors as input to LFA has so far always produced a better model (e.g., in fraction addition) than SimStudent or LFA alone.

Discussion

In this chapter, we focused on a key challenge in the KC approach to learner modeling, namely, that of accurately determining what the KCs are that learners acquire in a given task domain. While up until now, manual approaches to KC modeling have been prevalent, data-driven approaches have progressed to the point that they can at least be a useful supplement and perhaps even a substitute. The ways in which manual models based on intuition fall short provide interesting feedback on our intuitions about learning. Perhaps the examples given in the chapter illustrate that even experienced KC modelers tend to be over-optimistic about transfer. Perhaps the exercise of refining handmade models through data-driven, automated methods will eventually lead to a better general understanding when transfer is and is not likely to occur. Interestingly, the educational impact of KC modeling approaches will reach well beyond ITS development, as lessons learned through KC modeling can be applied to the design of a wide range of instructional materials, not just advanced learning technologies and ITSs. For example, many or all of the examples of discoveries about student learning given in the chapter could be applied to textbook and (non-tutored) homework assignments.

Given that we have described a range of tools for KC modeling, each with pros and cons, it may be useful to reflect on how these tools might feature in a realistic tutor development scenario. In an ideal scenario, an author might first build a tutor based on manual KC modeling. For example, the author might use an ITS authoring tool that supports KC modeling as an integral part of tutor development, such as CTAT. The author might build an example-tracing tutor, hand-write a production rule cognitive model, or automatically develop a production rule cognitive model by tutoring SimStudent. Ideally, the author would not rely on theoretical cognitive task analysis only but also do a substantial empirical task analysis (e.g., think-alouds and difficulty factors assessments), so as to increase the quality of the initial KC model.

However, even with empirical cognitive task analysis up front, there is still room for improvement through data-driven discovery from search over a larger space of tasks and models. After deploying the initial version of the tutor in a real educational setting, an author might use the data-driven tools described in this chapter to analyze the tutor log data. Specifically, the author evaluates how effective the tutor is in helping students learn the targeted KCs and explores ways in which the KC model can be refined using DataShop tools, using its learning curves display, fit indices for the AFM predictions, and the DataShop's performance profiler. For example, the author might investigate the causes of ragged learning curves using the performance profiler and try out a few model extensions by hand to see if AFM fit improves. In a more systematic approach, the author may use LFA to find a better model. Having found a better KC model (i.e., one that better matches the data about student learning), the author updates the tutor to improve its effectiveness ("closing the loop"). We emphasize that although the data-driven approaches such as use of DataShop tools, LFA, or SimStudent are helpful, in our opinion, they do not do away with the need for a careful cognitive task analysis at least to interpret the resulting data, but better yet toward creating a good KC model for use in the first version of the tutor.

Implications for GIFT

In this final section, we discuss implications for GIFT and other ITS authoring tools. We started the chapter by identifying three key challenges related to KC approaches to learner modeling: identifying psychologically accurate KCs, modeling the knowledge of individual students, and using KC models to individualize instruction. To support a KC approach to learner modeling, an ITS authoring tool would ideally address all three challenges. We illustrated how CTAT, in combination with the DataShop, successfully addresses these three challenges.

How could new authoring tools, such as GIFT, address these same challenges? Could they borrow from the CTAT/DataShop approach? We see some low-hanging fruit in particular with respect to the first challenge (creation/discovery/refinement of KC models). A good start is integration with the DataShop (Koedinger et al., 2010; Stamper & Koedinger, 2011). All that is needed is that tutors log student-tutor interactions in DataShop format (see http://pslcdatashop.org/dtd/). This integration is a good first step even before the authoring tool supports any kind of KC model. Even in the absence of any KC model within the tutoring system itself, the DataShop can be used for offline KC modeling based on tutor log data, which can lead to recommendations for improving the tutoring system. DataShop logging requires that the student interactions are divided into steps, which can involve successful attempts, unsuccessful attempts, and hint requests. VanLehn (2006) illustrates how interactions from many different tutoring systems can naturally be thought of as involving steps.

In addition, behavior graphs may be useful in GIFT and other authoring tools. Mapping out the solution space for tutored problems (as behavior graphs do) is useful for purposes of theoretical task analysis, as illustrated in the current chapter, and is compatible with a range of different types KC models.

Beyond this low-hanging fruit, the main challenge we see in supporting a KC approach in an authoring tool such as GIFT is in representing a particular type of KC model. Ideally, the authoring tool also makes it considerably easier for an author to build KC models of the targeted type in a range of application domains. A key related challenge is to the use of the KC model in the system's inner loop to interpret and assess student behavior. The commitment to support a particular type of KC model influences many aspects of the authoring tool and should be made early on in the tool's development. Different authoring tools address this challenge in their own way. CTAT supports two types of KC models, rule-based models and annotated behavior graphs. ASPIRE supports the authoring of constraints as key KCs with various tools (Mitrovic, Martin, Suraweera, Zakharov, et al., 2009).

With respect to the second and third challenges (using a KC model to track individual students' knowledge growth and individualizing instruction), innovative ITS authoring tools such as GIFT may include mechanisms or algorithms that enable tutors built with the tools to track over time how well individual learners do in terms of a given KC model. Also, they may offer a range of different ways of adapting the instruction based on such models of individual learners (the third challenge). By supporting easy customizability with respect to a range of options, authoring tools can do much to advance research into the benefits of different ways of individualizing instruction, a forte of ITSs.

References

Aleven, V. (2010). Rule-Based cognitive modeling for intelligent tutoring systems. In R. Nkambou, J. Bourdeau & R. Mizoguchi (Eds.), *Studies in Computational Intelligence: Vol. 308. Advances in intelligent tutoring systems* (pp. 33-62). Berlin, Heidelberg: Springer. doi:10.1007/978-3-642-14363-2_3

Aleven, V., McLaren, B. M., Sewall, J. & Koedinger, K. R. (2006). The cognitive tutor authoring tools (CTAT): Preliminary evaluation of efficiency gains. In M. Ikeda, K. D. Ashley & T. W. Chan (Eds.), *Proceedings of the 8th International Conference on Intelligent Tutoring Systems (ITS 2006)* (pp. 61-70). Berlin, Heidelberg: Springer-Verlag. doi:10.1007/11774303_7

Aleven, V., McLaren, B. M., Sewall, J. & Koedinger, K. R. (2009). A new paradigm for intelligent tutoring systems: Example-Tracing tutors. *International Journal of Artificial Intelligence in Education*, *19*(2), 105-154.

Aleven, V., Roll, I. & Koedinger, K. R. (2012). Progress in assessment and tutoring of lifelong learning skills: An intelligent tutor agent that helps students become better help seekers. In P. J. Durlach & A. M. Lesgold (Eds.), *Adaptive technologies for training and education* (pp. 69-95). New York: Cambridge University Press.

Anderson, J. R. (1993). *Rules of the mind*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Anderson, J. R., Corbett, A. T., Koedinger, K. R. & Pelletier, R. (1995). Cognitive tutors: Lessons learned. *The Journal of the Learning Sciences*, *4*(2), 167-207.

Arroyo, I., Beck, J., Woolf, B. P., Beal, C. R. & Schultz, K. (2000). Macroadapting animalwatch to gender and cognitive differnces with respect to hint interactivity and symbolism. In G. Gauthier, C. Frasson & K. VanLehn (Eds.), *Proceedings of the 5th International Conference on Intelligent Tutoring Systems* (ITS 2000) (pp. 574-583). Berlin: Springer Verlag.

Baker, R. S. J. d., Corbett, A. T. & Koedinger, K. R. (2007). The difficulty factors approach to the design of lessons in intelligent tutor curricula. *International Journal of Artificifial Intelligence and Education*, *17*(4), 341-369.

Baker, R. S. J. D., Corbett, A. T., Koedinger, K. R., Evenson, S., Roll, I., Wagner, A. Z., . . . Beck, J. E. (2006). Adapting to when students game an intelligent tutoring system. In *ITS'06: Proceedings of the 8th International Conference on Intelligent Tutoring Systems (ITS 2006)* (pp. 392-401). Berlin, Heidelberg: Springer-Verlag. doi:10.1007/11774303_39

Baker, R. S. J. D., Goldstein, A. B. & Heffernan, N. T. (2011). Detecting learning moment-by-moment. *International Journal of Artificial Intelligence in Education*, *21*(1-2), 5-25. doi:10.3233/JAI-2011-015

Bull, S. & Kay, J. (2007). Student models that invite the learner in: The SMILI: () Open learner modelling framework. *International Journal of Artificial Intelligence in Education*, *17*(2), 89 - 120. Retrieved from http://search.ebscohost.com/login.aspx?direct=true&db=psyh&AN=2008-01222-001&site=ehost-live

Cen, H., Koedinger, K. & Junker, B. (2006). Learning factors analysis--a general method for cognitive model evaluation and improvement. In M. Ikeda, K. D. Ashley & T. W. Chan (Eds.), *Proceedings of the 8th International Conference on Intelligent Tutoring Systems (ITS 2006)* (pp. 164-175). Berlin: Springer.

Cen, H., Koedinger, K. R. & Junker, B. (2007). Is over practice necessary?-improving learning efficiency with the cognitive tutor through educational data mining. In R. Luckin, K. R. Koedinger & J. Greer (Eds.), *Proceedings of the 13th International Conference on Artificial Intelligence in Education* (pp. 511-518). Amsterdam: IOS Press.

Conati, C. & Vanlehn, K. (2000). Toward computer-based support of meta-cognitive skills: A computational framework to coach self-explanation. *International Journal of Artificial Intelligence in Education*, *11*(4), 389-415. Retrieved from Google Scholar.

Conati, C., Gertner, A. & Vanlehn, K. (2002). Using bayesian networks to manage uncertainty in student modeling. *User Modeling and User-adapted Interaction*, *12*(4), 371-417.

Corbett, A., McLaughlin, M. & Scarpinatto, K. C. (2000). Modeling student knowledge: Cognitive tutors in high school and college. *User Modeling and User-Adapted Interaction*, *10*, 81-108.

Corbett, A. T. & Anderson, J. R. (1995). Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Modeling and User-Adapted Interaction*, *4*(4), 253-278.

Falmagne, J. -C., Koppen, M., Villano, M., Doignon, J. -P. & Johannesen, L. (1990). Introduction to knowledge spaces: How to build, test and search them. *Psychological Review*, *97*(2), 201-224.

Y. Gong, J. E. Beck & N. T. Heffernan (2011). How to construct more accurate student models: Comparing and optimizing knowledge tracing and performance factor analysis. *International Journal of Artificial Intelligence in Education, 21*(1-2), 27–46.

Harrer, A., McLaren, B. M., Walker, E., Bollen, L. & Sewall, J. (2006). Creating cognitive tutors for collaborative learning: Steps toward realization. *User Modeling and User-Adapted Interaction*, *16*(3-4), 175-209. doi:10.1007/s11257-006-9007-4

Koedinger, K. R. & Aleven, V. (2007). Exploring the assistance dilemma in experiments with cognitive tutors. *Educational Psychology Review*, *19*(3), 239-264.

Koedinger, K. R., Aleven, V., Heffernan, N., McLaren, B. & Hockenberry, M. (2004). Opening the door to non-programmers: Authoring intelligent tutor behavior by demonstration. In J. C. Lester, R. M. Vicario & F. Paraguaçu (Eds.), *Proceedings of Seventh International Conference on Intelligent Tutoring Systems, ITS 2004* (pp. 162-174). Berlin: Springer.

Koedinger, K. R., Baker, R. S. J. d., Cunningham, K., Skogsholm, A., Leber, B. & Stamper, J. (2010). A data repository for the EDM community: The PSLC datashop. In S. Ventura, C. Romero, M. Pechenizkiy & R. S. J. d. Baker (Eds.), *Handbook of educational data mining.* Boca Raton, FL: CRC Press.

Koedinger, K. R., Corbett, A. T. & Perfetti, C. (2012). The knowledge-learning-instruction framework: Bridging the science-practice chasm to enhance robust student learning. *Cognitive Science*, *36*(5), 757-798. doi:10.1111/j.1551-6709.2012.01245.x

Koedinger, K.R. & McLaughlin, E.A. (2010). Seeing language learning inside the math: Cognitive analysis yields transfer. In S. Ohlsson & R. Catrambone (Eds.), *Proceedings of the 32nd Annual Conference of the Cognitive Science Society* (pp. 471-476). Austin, TX: Cognitive Science Society.

Koedinger, K. R., McLaughlin, E. A. & Stamper, J. C. (2012). Automated student model improvement. In K. Yacef, O. Zaïane, A. Hershkovitz, M. Yudelson & J. Stamper (Eds.), *Proceedings of the 5th International Conference on Educational Data Mining* (pp. 17-24). International Educational Data Mining Society, www.educationaldatamining.org.

Koedinger, K. R. & Nathan, M. J. (2004). The real story behind story problems: Effects of representations on quantitative reasoning. *The Journal of the Learning Sciences*, *13*(2), 129-164.

Koedinger, K.R., Stamper, J.C., McLaughlin, E.A. & Nixon, T. (in press). Using data-driven discovery of better student models to improve student learning. To appear in *Proceedings of the 16th International Conference on Artificial Intelligence in Education*.

Lee, J. I. & Brunskill, E. (2012). The impact on individualizing student models on necessary practice opportunities. In K.Yacef, O.R.Zaïane, A. Hershkovitz, M. Yudelson & J. C. Stamper (Eds.), *Proceedings of the 5th International Conference on Educational Data Mining,* (pp. 118–125). International Educational Data Mining Society, www.educationaldatamining.org.

Li, N., Matsuda, N., Cohen, W. W. & Koedinger, K. R. (2011). A machine learning approach for automatic student model discovery. In M. Pechenizkiy, T. Calders, C. Conati, S. Ventura, C. Romero & J. Stamper (Eds.), *Proceedings of the 4th International Conference on Educational Data Mining* (pp. 31-40). International Educational Data Mining Society, www.educationaldatamining.org.

Long, Y. & Aleven, V. (2013). Supporting students' self-regulated learning with an open learner model in a linear equation tutor. To appear in *the Proceedings of the 16th International Conference on Artificial Intelligence in Education (AIED 2013).*

Lovett, M., Meyer, O. & Thille, C. (2008). JIME-The open learning initiative: Measuring the effectiveness of the OLI statistics course in accelerating student learning. *Journal of Interactive Media in Education*, *2008*(1).

Lovett, M. C. (1998). Cognitive task analysis in the service of intelligent tutoring system design: A case study in statistics. In B. P. Goettle, H. M. Halff, C. L. Redfield & V. J. Shute (Eds.), *Intelligent Tutoring Systems, Proceedings of the Fourth International Conference* (pp. 234-243). Berlin: Springer Verlag.

Mathan, S. A. & Koedinger, K. R. (2005). Fostering the intelligent novice: Learning from errors with metacognitive tutoring. *Educational Psychologist*, *40*(4), 257-265.

Mitrovic, A. & Martin, B. (2004). Evaluating adaptive problem selection. In P. M. E. de Bra & W. Nejdl (Eds.), *Proceedings of the Third International Conference on Adaptive Hypermedia and Adaptive Web-based Systems, AH 2004* (Vol. 3137, pp. 185-194). New York : Springer-Verlag .

Mitrovic, A. & Martin, B. (2007). Evaluating the effect of open student models on self-assessment. *International Journal of Artificial Intelligence in Education*, *17*(2), 121-144.

Mitrovic, A., Martin, B., Suraweera, P., Zakharov, K., Milik, N., Holland, J. & Mcguigan, N. (2009). ASPIRE: An authoring system and deployment environment for constraint-based tutors. *International Journal of Artificial Intelligence in Education, 19*(2), 155-188

Mitrovic, A., Mayo, M., Suraweera, P. & Martin, B. (2001). Constraint-based tutors: A success story. In L. Monostori, J. Váncza & M. Ali (Eds.), *Proceedings 14th International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems (IEA/AIE 2001)* (pp. 931-940). Berlin Heidelberg: Springer.

Muldner, K. & Conati, C. (2007). Evaluating a decision-theoretic approach to tailored example selection. In M. Veloso (Ed.), *Proceedings of the 20th International Joint Conference on Artificial Intelligence, IJCAI 2007* (pp. 483-488). San Francisco, CA: Morgan Kaufmann.

Murray, T., Blessing, S. & Ainsworth, S. (2003). *Authoring tools for advanced technology learning environments: Toward cost-effective adaptive, interactive and intelligent educational software.* Amsterdam: Kluwer Academic Publishers.

Newell, A. & Simon, H. A. (1972). *Human problem solving.* Englewood Cliffs, NJ: Prentice-Hall.

Roll, I., Aleven, V., McLaren, B. M. & Koedinger, K. R. (2011). Improving students' help-seeking skills using metacognitive feedback in an intelligent tutoring system. *Learning and Instruction*, *21*(2), 267-280.

Salden, R. J. C. M., Aleven, V. A., Renkl, A. & Schwonke, R. (2009). Worked examples and tutored problem solving: Redundant or synergistic forms of support? *Topics in Cognitive Science*, *1*(1), 203-213.

Spada, H. & McGaw, B. (1985). The assessment of learning effects with linear logistic test models. In S. E. Embretson (Ed.), *Test design: Developments in psychology and psychometrics* (pp. 169-194). Orlando, FL: Academic Press.

Stamper, J. C. & Koedinger, K. R. (2011). Human-machine student model discovery and improvement using datashop. In G. Biswas, S. Bull, J. Kay & T. Mitrovic (Eds.), *Proceedings of the 15th International Conference on Artificial Intelligence in Education (AIED 2011)* (pp. 353-360). Berlin, Heidelberg: Springer-Verlag.

VanLehn, K. (2006). The behavior of tutoring systems. *International Journal of Artificial Intelligence in Education*, *16*(3), 227-265.

VanLehn, K. (2011). The relative effectiveness of human tutoring, intelligent tutoring systems, and other tutoring systems. *Educational Psychologist*, *46*(4), 197-221.

VanLehn, K., Lynch, C., Schulze, K., Shapiro, J. A., Shelby, R., Taylor, L., . . . Wintersgill, M. (2005). The Andes physics tutoring system: Lessons learned. *International Journal of Artificial Intelligence in Education, 15*(1), 147-204.

Waalkens, M., Aleven, V. & Taatgen, N. (2013). Does supporting multiple student strategies lead to greater learning and motivation? Investigating a source of complexity in the architecture of intelligent tutoring systems. *Computers & Education*, *60*(1), 159-171. doi:10.1016/j.compedu.2012.07.016

Wilson, M. & De Boeck, P. (2004). Descriptive and explanatory item response models. In P. De Boeck & M. Wilson, (Eds.) *Explanatory item response models: A generalized linear and nonlinear approach* (pp. 43-74). New York: Springer-Verlag.

CHAPTER 16 –Towards Learner Models based on Learning Progressions (LPs) in DeepTutor

Vasile Rus, William Baggett, Elizabeth Gire, Don Franceschetti, Mark Conley, and Arthur Graesser
The University of Memphis

Introduction

Learner modeling is a central component in any tutoring system that also claims to be intelligent. A learner model is a set of learner characteristics that impact learning, and therefore, is important for adapting instruction to each individual learner and thereby maximize learning. Indeed, research has shown that learners' diverse backgrounds require tailoring instruction to each individual learner or, more realistically in classroom settings, to homogeneous groups of learners (Corcoran, Mosher & Rogat, 2009). It should be noted that a major advantage of computer-based instruction, including ITSs, over classroom instruction is the potential of offering tailored instruction to each and every learner. These systems presumably can be scaled up in the sense that once developed for one user they could be easily replicated or modified to handle many users. Learner modeling is a particular case of user modeling, which constitutes a central component in any user adaptive system. User adaptive systems tailor their behavior to the individual user's characteristics in order to optimize the value of their function with respect to the user (Conati, Gertner & VanLehn, 2002; VanLehn, 2006). We focus here on user models in ITSs, i.e., on learner models.

The learner model plays a role in both the outer and inner loop of ITSs (VanLehn, 2006). The outer loop handles macro-adaptivity in the sense that it selects tasks and other instructional materials to present to the learner. The outer loop also selects a mode for the task, e.g., demonstrating a step-by-step solution to the task or just providing hints. The inner loop manages the tutor-tutee interaction within a task by monitoring the learners' actions while she is working on an instructional task. The inner loop handles micro-adaptivity.

We present in this chapter a novel approach to learner modeling that enables improved macro- and micro-adaptivity. We emphasize here the impact of the new approach on macro-adaptivity. The novel approach relies on a research framework, called Learning Progressions (LPs), developed recently by the science education research community as a way to increase adaptivity in traditional instruction. Indeed, "assessment for learning" (Black & William, 1998) has been a focus in this community for more than a decade (National Research Council, 2001, 2005, 2007). This effort led to the emergence of the framework of LPs, defined as "descriptions of the successively more sophisticated ways of thinking about an idea that follow one another as students learn" (NRC, 2005, 2007). Corcoran, Mosher, and Rogat (2009) state "progressions can play a central role in supporting the needed shift toward adaptive instruction as the norm of practice in American schools." Importantly, LPs provide a promising means to organize and align content, instruction, and assessment strategies to give students the opportunity to develop deep and integrated understanding of science ideas. The time is ripe for the ITS community to integrate in computer tutors such advances in assessment and instruction proposed by the assessment and science education research communities in order to increase ITSs' adaptivity and, in turn, their effectiveness at inducing learning gains in learners.

Our dialogue-based ITS DeepTutor (Rus et al., to appear) incorporates the framework of LPs (National Research Council, 2007) as a way to improve assessment and better tailor instruction to each individual learner. DeepTutor is under development as of this writing, which explains the preliminary flavor of the ideas presented in this chapter. Because DeepTutor is a dialogue-based ITS, the main form of interaction in DeepTutor is tutorial dialogue that mimics interaction between human tutors and learners. Deep natural language processing technologies are also needed to accurately assess students' level of understanding while interacting with DeepTutor. In fact, the quality of the algorithms for dialogue and language processing has a direct impact on other core ITS tasks such as feedback generation. An authoring tool that allows us to explore and design algorithms for deep natural language processing has been developed (for more information about the Semantic Similarity, or simply SEMILAR, toolkit, see www.semanticsimilarity.org).

Based on (1) LPs that promote deep and integrated understanding of science and (2) deep natural language processing algorithms (as well as several other novel aspects of DeepTutor such as advanced tutorial strategies which we do not discuss here), DeepTutor is expected to provide better assessment and, in turn, better adapt instruction to each individual learner. This will lead to learning gains beyond the interaction plateau, i.e., the hypothesis that as interactivity of tutors increases, their effectiveness plateaus (VanLehn, 2011). VanLehn's interactivity plateau finding calls for ITS researchers to propose qualitative advances that can increase the learning effectiveness beyond the interactivity plateau. Indeed, the proposed advances in DeepTutor are meant to address this challenge by proposed qualitative shifts for core ITS components such as the domain and student modeling components through the use of LPs.

The rest of this chapter is organized as follows. After a brief general discussion of learner modeling in ITSs, we describe the framework of LPs. Then, we describe general ideas about the development and validation of LPs and illustrate how this process is implemented in DeepTutor, the first ITS based on LPs. The role of LPs for macro-adaptation in our DeepTutor project is then highlighted. We conclude with a discussion about how GIFT can accommodate LP-driven ITSs.

Learner Modeling in ITSs

ITS researchers have investigated and integrated learner models in their systems to increase the effectiveness of these systems (Conati, Gertner & VanLehn, 2002; Corcoran, Mosher & Rogat, 2009; Lintean, Rus & Azevedo, 2012; Sottilare et al., 2012; VanLehn, 2006). Although other learner characteristics such as emotions have been considered recently, researchers and developers of ITSs have focused primarily on learners' performance with respect to the target domain, i.e., knowledge assessment, to guide adaptivity.

Different ITSs employ different approaches to the problem of learning modeling, including knowledge tracing in the Cognitive Tutor (Anderson et al., 1995), constraint-based modeling (Mitrovic, Martin & Suraweera, 2007), and the Expectation-Misconception approach used in AutoTutor (Graesser, Rus, D'Mello & Jackson, 2008). In all of these models, the emphasis is on supporting micro-adaptivity. Attempts to handle macro-adaptivity have been modest. The Student Modeling Approach for Responsive Tutoring (SMART) student model (Shute, 1995) uses regression equations to estimate students' mastery of each curriculum element. Conati, Gertner, and VanLehn (2002) describe the learner model used in their Andes system that tutors students on Newtonian physics. The model is based on Bayesian networks in which explicit domain-general and context-specific rules are encoded. Conati and colleagues describe the use of the model for micro-adaptivity during example studying and problem-solving, but they do not report any use for macro-adaptivity. The mathematics tutor ALEKS (Doignon & Falmagne, 1999) relies on the knowledge space theory (KST) for domain modeling. KST relies on the precedence relation that is evident for some domains like mathematics where a student must learn a concept A before learning another that relies on A. A student's knowledge state is the set of items mastered. Not all subsets of items are feasible knowledge states due to the precedence relations among items. Based on assessment, a student's knowledge state is inferred, which guides instruction. Learning occurs on the outer fringe of the knowledge state that is the immediate successor state in the knowledge structure of the domain. The knowledge structure is the collection of all possible states. ALEKS offers macro-adaptivity but the organization of the domain is based on a logical organization provided by experts according to intrinsic dependencies among concepts in a domain. There is no emphasis on the developmental milestones students pass while learning a topic. KST may be suitable for domains where precedence relations are well defined such as mathematics but for domains where there is no such clear structure KST is less useful. An interesting research question is about how KST for well-defined domains can be transformed into or combined with the LP framework.

There are three general challenges with deriving knowledge assessment learner models for each individual student based on her actions during learning: (1) determining how much activity from an individual learner the system must monitor and analyze to reliably estimate the parameters of the learner model, (2) keeping the parameters up to date while the student's level of understanding of a domain evolves, and (3) representing and handling misconceptions. The knowledge assessment learner models encode primarily what the students are supposed to learn, i.e., true knowledge, while not embedding explicitly alternative conceptions. These alternative conceptions, or misconceptions, are handled separately. Novel solutions based on LPs can better address these issues as shown next in the context of our DeepTutor ITS.

We use the framework of LPs as a paradigm shift in learner modeling for ITSs. LPs model and organize domain knowledge based on what is known about how learners actually progress through that content as opposed to a logical decomposition of the content by a domain expert. LPs explicitly consider the alternative conceptions, mapping how they developmentally emerge. This learner-centered organization of content enables improved macro-adaptation and micro-adaptation. We focus here on the role of LPs in supporting better macro-adaptation. Coupled with appropriate assessment instruments and instructional strategies, LPs offer the possibility of qualitative shifts in learner modeling and learner experiences with ITSs.

The Framework of Learning Progressions

The National Research Council (NRC) (2001) report called for better descriptions of how students learn based on models of cognition and learning. Based on such descriptions of how students learn, "assessments can be designed to identify current student thinking, likely antecedent understandings, and next steps to move the student toward more sophisticated understandings" (NRC, 2001, p. 182). This was basically a call for developing LPs (Corcoran, Mosher & Rogat, 2009). The term "learning progressions" was first used in a subsequent NRC report (2005). According to Corcoran, Mosher & Rogat (2009), LPs in science are "empirically grounded and testable hypotheses about how students' understanding of, and ability to use, core scientific concepts and explanations and related scientific practices grow and become more sophisticated over time, with appropriate instruction" (p.8).

Corcoran, Mosher, and Rogat (2009) also mention as major conceptual contributions (although they did not use the term LP): (1) the work of Lehrer and Shauble (2000) on model-based reasoning in science; and (2) research by Valverde and Schmidt's (1997) on the relationship between content and structure of curricula and student achievement. It is also worth mentioning the work of Roberts,Wilson, and Draney, (1997) on "construct maps" and work on progress maps by Masters and Fosters (1996). Wilson and colleagues and Masters and Fosters pushed for a concentrated effort on improving assessment to guide instruction. Work on learning trajectories in mathematics (Driver, Leach, Scott & Wood-Robinson, 1994) as well as work on cognitive development maps (Baxter & Junker, 2001) further contributed to the development of LPs.

LPs can be viewed as incrementally more sophisticated ways to think about an idea that emerge naturally while students move toward expert-level understanding of the idea (Duschl et al., 2007). LPs define qualitatively different levels of understanding of big ideas. The levels can be sequentially related or the relation could be more complex. For instance, topic A may develop the ideas from a less sophisticated topic B but also connect to other topics. LPs model how students develop deep and integrated understanding of complex science topics. The emphasis on increased sophistication (i.e., depth) and integrated understanding of ideas sets LPs apart from classroom instruction and assessment, which are typically based on curriculum materials that follow local, state, and national standards. These standards tend to support compartmentalized understanding and shallow coverage of a broad range of topics instead of a deep, integrated understanding of few key ideas (Schmidt, Wang & McKnight, 2005; Stevens, Delgado & Krajcik, 2009). The bottom line is that LPs adopt a *learner-centric* view of a topic, modeling students' successful paths towards mastery as opposed to paths prescribed by domain experts following a logical decomposition of big ideas. The logical decomposition provided by experts could be useful as a starting point but then needs be reorganized based on evidence of how students actually develop mastery of the big ideas. These actual paths must be documented and guide instruction. While some paths towards mastery defined by domain experts may be successfully followed by some students, some other students may follow other paths, i.e., might be more responsive to different topic sequences as well as instructional tasks that help their conceptual change.

LPs are organized in levels of understandings that reflect major milestones in learners' journey towards mastery. The lower level, called the Lower Anchor, represents naïve thinking that usually novices hold. The top level, called the Upper Anchor, represents the mastery/expert level of understanding. If targeting a particular population, e.g., high school students, the Lower Anchor would coincide with the Upper Anchor of grade 8 students. Such an assumption is not always practical, as we learned during the development of our DeepTutor system, because not all students entering high school are at the Upper Anchor of grade 8. Furthermore, if someone develops an instructional intervention such as DeepTutor to be used by learners of all ages then the Lower Anchor should be specified accordingly, i.e., to reflect the lowest possible level of understanding. The anchoring of the Upper Level in an LP raises the issue of whether standards should indicate what the best and brightest students can achieve in a particular grade or whether standards should consider what average students can do (Corcoran et al., 2009). It is beyond the scope of this chapter to discuss this issue. In our work, we use standards to specify the Upper Anchor for a particular grade level. Our true Upper Anchor specifies the true and strongest scientific conceptions.

Open Issues

While there are many commonalities and a general understanding of what LPs are, some aspects of LPs are open for debate and interpretation and vary from one developer to another. For instance, the span of an LP can vary from one instructional unit that is covered over several weeks during a semester in a particular grade to covering multiple grades, e.g., 6–8 years of instruction. Alonzo and Steedle (2009) differentiate between broad LPs and small LPs that cover in more detail a particular idea or construct in the sense proposed by Wilson in his Structured Construct Model (SCM; Wilson, 2009). That is, broad LPs may cover the development of a set of big ideas with each big idea constituting a small LP. The broad LP has the advantage of providing the big picture: the big ideas and how they develop across grades as well as the inter-dependencies among these big ideas. The smaller LPs provide sufficient detail such that instructors can use them to track students' progress over instructional units and guide their instruction.

Another varying aspect of LPs is their granularity. Fewer levels described in summary core ideas means a more general LP and also more reliable diagnostics. However, finer-grained LPs offer a more sensitive instrument to measure progress, have more explanatory power, and can be more useful in guiding instruction. A related contentious aspect of LPs is their relationship with curriculum and instruction. Some developers do consider instruction as an integral part of LPs (Songer, Kelcey & Gotwals, 2009) while others develop LPs independent of instruction (Mohan, Chen & Anderson, 2009). The approach adopted affects the LP validation process.

We conclude this section by noting that research on LPs is thriving with many LPs being developed (Alonzo & Steedle, 2009; Songer, Kelcey & Gotwals, 2009; Jin & Anderson, 2012; Neumann, Viering, Boone & Fischer, 2012; Johnson & Tymms, 2011), many LPs conferences and other events being organized, and LPs being adopted by various states. Of note are the LPs adopted by the state of Massachusetts (Foster & Wiser, 2012).

The Development and Validation of LPs in Deeptutor

We now describe the structure and process of developing and validating LPs. LP development is driven by assessment, i.e., what gets measured and what counts as evidence for learning, and should include the following:

1. Content and conceptions (student thinking) – what gets measured. That is, LPs must specify what students need to know and the various alternative conceptions they may have.

2. Learning performances, which are the “operational definitions of what learners’ understanding and skills would look like at each of these stages of progress” (Corcoran et al., 2009) – what counts as evidence of level of understanding.

3. The Upper Anchor describes the set of knowledge and skills students are expected to have at the end of the progression. These expectations can be found in state or national science standards and recent learning research on the subject matter.

4. Progress levels or steps of achievement (intermediate levels) include both correct and incorrect/weak conceptions that learners consistently use to reason about phenomena in a given domain. Some of the correct conceptions are held early on. Some weak conceptions (or misconceptions) can be extremely persistent across many progress levels.

5. The Lower Anchor describes student conceptions at entry level, i.e., when they start learning about something.

6. Assessments that measure student understanding of the key concepts or practices.

7. Purposeful curriculum and instruction that mediates targeted student outcomes (Duschl, 2011; Corcoran et al., 2009).

We describe next the process we followed in developing the Force and Motion LP in our DeepTutor project. We adopted a design-based research iterative process that allowed us to develop and validate the progressions (Mohan, Chen & Anderson, 2009; Stevens et al., 2009). The process first conjectured a hypothetical learning progression (HLP) and then derived an empirical progression (EP), which is a refined version of the HLP. After several iterations of refinements and empirical validations, the HLP became an empirically tested LP. We used the following three criteria to guide our validation process: conceptual coherence, compatibility with current research, and empirical support from real student data (Mohan et al., 2009). The development of the initial HLP presupposes defining the ideas and concepts to learn at an appropriate level of detail while considering recent cognitive and education research that could provide insights about potential challenges students face or typical prior knowledge students may be expected to have.

The result of our LP design effort in the DeepTutor project is a broad Newtonian physics LP structured in seven strands, or smaller LPs in the sense of Alonzo and Steedle (2009). We have one small LP for each of seven themes or big ideas in Newtonian physics: Kinematics, Force and Motion (linear motion), Mass and Motion, Free-Fall new Earth, Newton's Third Law, Vectors and Motion (motion in two dimensions), and Circular Motion. Each strand is organized in a number of levels. The number of levels in the broad LP varies from strand to strand, e.g., the Mass and Motion strand has three levels while the Free-Fall near Earth strand has seven levels. Levels are not equivalent across strands, i.e., level 2 in the Mass and Motion strand is not equivalent to level 2 in the Free-Fall strand. The strands, in turn, are ordered in terms of their complexity and prerequisite requirements. For instance, understanding the basic concepts of position, velocity, and acceleration covered in Kinematics is needed before studying Newton's second law in the Force and Motion strand. That is, we have a 2-D broad LP in which one dimension illustrates students' level of understanding and the other the complexity and interdependencies among the big themes (see the left side of Figure 16-1). The broad LP can also be regarded as a one-strand LP with seven levels of understanding, each level corresponding to one of the big physics themes/topics (see Figure 16-1, right side). This HLP has been validated based on data collected from high school students. The HLP is now an EP. Several more iterations of refinement are needed to obtain an empirically tests LP.

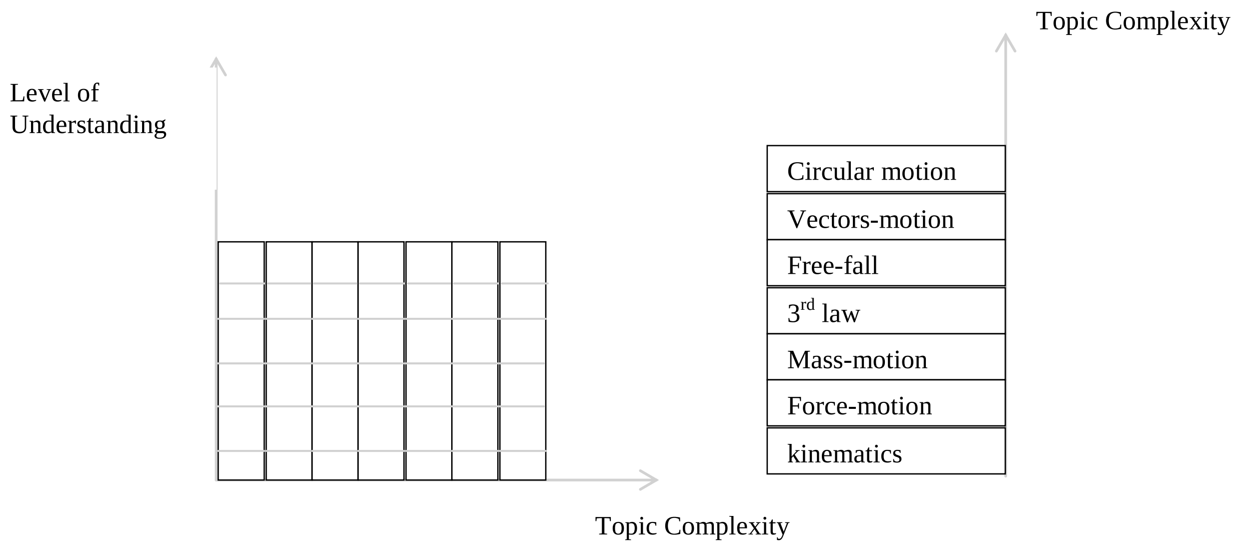


Figure 16-1. The Newtonian physics LP: 2-D depiction (left) vs. 1-D, broad LP depiction (right)

Our LP works in close relationship with a set of assessment items that allows the instructor, in our case DeepTutor, to place the student somewhere in the LP. Based on where the student is placed in the LP, instructional tasks and materials that are appropriate for that level of understanding are assigned to the student. Because the LP levels are so designed to encode the most successful paths to mastery followed by students, the LPs greatly facilitate the selection of tasks to be given to a particular student, i.e., LPs enable macro-adaptation of instruction. Tasks and materials associated with a level in the LP are so designed to help students improve their understanding and move up the LP hierarchy.

It should be noted that tasks based on an LP can be sequenced in various ways, leading students on different learning trajectories. For instance, drilling tasks, which offer training on one big theme modeled by one LP strand, will more likely help students move up the level of understanding within that strand while not making progress on other big ideas. For some strands, e.g., Newton's 3rd law, for which the correct answer to many problems is the same (the tasks are isomorphic at some degree), a repetitive drilling strategy is problematic as students may learn the jingle ("for every action there is an equal and opposite reaction") after seeing the solution to a few problems and just recite the jingle when prompted for solutions to subsequent tasks without actually developing a deep understanding of Newton's 3rd law. Smarter sequencing of problems must be adopted as isomorphic problems lead to copying and therefore shallow learning (VanLehn, 2011; Renkl, 2002).

DeepTutor infers learners' levels of understanding of the target domain using both summative and formative or embedded assessment. Summative assessment consists of pre- and post-tests. The inferred levels of understanding based on the pre-test are continuously updated based on learners' performance while working on various tasks with the system, i.e., through formative assessment. Formative assessment is seamless as learners are assessed as they work on a problem without being aware of this type of assessment. Formative assessment directly and immediately impacts micro-adaptation while indirectly impacting macro-adaptation through the continuous update of the learner model. For instance, learners who embrace many misconceptions or confuse concepts such as velocity and acceleration or do not understand the fine difference between velocity and speed will have their level of understanding updated accordingly even though they might have done well on the pre-test. We are moving toward a cloud-model of student assessment in which we allow students to simultaneously hold in their minds different models of reasoning or levels of understanding of a target domain (Rus et al., in press). Students will activate one model or level with a certain probability distribution. Improving the alignment between assessment and instruction using LPs is a major feature in DeepTutor, which is expected to increase the adaptivity of the system and in turn improve students' learning experience and learning gains.

Discussion with Respect to the Generalized Intelligent Framework for Tutoring

GIFT (Sottilare, Brawner, Goldberg & Holden, 2012) aims at providing a standard that unifies and streamlines ITS development efforts. Modularity, communication among modules, and separation of code from content are the driving principles of GIFT. Furthermore, GIFT supports a service-oriented architecture to facilitate distributed and mobile learning. The GIFT framework includes the following major modules: sensors module, user module, pedagogy module, and domain module. All these major modules are domain independent except the latter. The user module includes the learner model among models for other users such as trainer, expert, and designer. The domain module performs assessment functions such that the only domain-dependent module is the domain module.

As a web service, DeepTutor is accessible by any learner, anytime, anywhere. Indeed, DeepTutor can be accessed through an HTML5-compatible browser from desktop computers and mobile devices such as smartphones and tablets. As such, it ought to be generally compatible with a generic framework such as GIFT. However, integrating DeepTutor into the GIFT framework will require some important modifications. For one thing, although DeepTutor also employs a a modular design, the modules are organized differently than those in GIFT. For instance, assessment is a separate, independent module from the domain module. More modules mean that the development and management of these modules, especially when scaling up, are more feasible. Assessment is a big topic as well as is domain modeling (knowledge acquisition and representation), which may suggest that they have to be separate modules for scalability purposes. Also, because DeepTutor relies on LPs there is an intrinsic relation between the domain knowledge and learner model. The learner's level of understanding as modeled by the LPs is not recorded in the persistent learner model, stored in a database to be updated and used over time, as a set of learner characteristics and their values (as suggested by the GIFT developers) but rather as pointers to LP levels. This arrangement makes the learner and domain modules tightly connected. If the two modules should be decoupled as suggested by GIFT, then the learner module becomes domain-dependent. Indeed, the characteristic-value learner model implies that the learner model should include domain-specific information because some of the learner characteristics reflect learner's performance in a target domain, e.g., how well the student understands Newton's 3rd law. That is, the learner module needs to include domain-specific performance measures that need be traced. A potential solution would be for GIFT to specialize its proposed specifications for various types of ITSs, similar to specialization principles outlined in Pavlik, Mass, Olney, and Rus (2012).

Acknowledgments

This research was supported in part by Institute for Education Sciences under awards R305A100875. Any opinions, findings, and conclusions or recommendations expressed in this material are solely the authors' and do not necessarily reflect the views of the sponsoring agencies.

References

Alonzo, A. C. & Steedle, J. T. (2009). Developing and assessing a force and motion learning progression. Science Education, 93, 389-421.

Anderson, J. R., Corbett, A. T., Koedinger, K. R. & Pelletier, R. (1995). Cognitive tutors: Lessons learned. *Journal of the Learning Sciences*, *4*, 167-207.

Baxter, G. and Junker, B. W. (2001). Designing Developmental Assessments: A Case Study in Proportional Reasoning. Paper presented at the Annual Meeting of the National Council of Measurement in Education, April 2001, Seattle, WA.

Black, P. & Wiliam, D. (1998). Assessment and classroom learning. Assessment in Education, 5, 7–74.

Conati C., Gertner A. and VanLehn K. (2002). Using Bayesian Networks to Manage Uncertainty in Student Modeling . Journal of User Modeling and User-Adapted Interaction, vol. 12(4), p. 371-417.

Corcoran, T., Mosher, F.A. & Rogat, A. (2009). Learning progressions in science: An evidencebased approach to reform. Consortium for Policy Research in Education Report #RR-63. Philadelphia, PA: Consortium for Policy Research in Education.

Doignon, J.P. & Falmagne, J. C. (1999). *Knowledge Spaces*. Berlin, Germany: Springer.

Driver, R., Leach, J., Scott, P. & Wood-Robinson, C. (1994). Young people's understanding of science concepts: Implications of cross-age studies for curriculum planning. Studies in Science Education, 24, 75–100.

Duschl, R., Maeng, S. & Sezen, A. (2011). Learning progressions and teaching sequences: a review and analysis, Studies in Science Education, 47:2, 123-182.

Foster, J. & Wiser, M., (2012). The Potential of Learning Progressions Research to Inform the Design of State Science Standards. In A.C. Alonzo & A.W. Gotwals (Eds.) *Learning Progressions in Science: Current Challenges and Future Directions* (pp. 436). Sense Publishers.

Graesser, A. C., Rus., V., D'Mello, S., K., and Jackson, G. T. (2008). AutoTutor: Learning through natural language dialogue that adapts to the cognitive and affective states of the learner. In D. H. Robinson & G. Schraw (Eds.), Current perspectives on cognition, learning and instruction: Recent innovations in educational technology that facilitate student learning (pp. 95–125). Information Age Publishing.

Jin, H. and Anderson, C. W. (2012), A learning progression for energy in socio-ecological systems. J. Res. Sci. Teach., 49: 1149–1180.

Johnson, P. & Tymms, P. (2011). The Emergence of a Learning Progression in Middle School Chemistry. J. Res. Sci. Teach. 48(8): 849-877.

Lintean, M., Rus, V. & Azevedo, R. (2012). Automatic Detection of Student Mental Models during Prior Knowledge Activation in MetaTutor. International Journal of Artificial Intelligence in Education, 21(3), pp. 169-190.

Lehrer, R. & Schauble, L. (2000). Modeling in mathematics and science. In R. Glaser (Ed.), Advances in instructional psychology: Education design and cognitive science (vol. 5, pp. 101-169). Mahway, NJ: Lawrence Erlbaum Associates.

Mitrovic, A., Martin, B. & Suraweera, P. (2007). Intelligent tutors for all: The constraint-based approach. *IEEE Intelligent Systems*, *22*(4), 38-45.

Mohan, L., Chen, J. &Anderson,W.A. (2009). Developing a multi-year learning progression for carbon cycling in socio-ecological systems. Journal of Research in Science Teaching, 46, 675–698.

National Research Council. (2001). Knowing what students know: The science and design of educational assessment. (J.W. Pellegrino, N. Chudowsky & R. Glaser, Eds.). Washington: National Academy Press.

National Research Council. (2005). Systems for state science assessment. (M.R. Wilson & M.W. Bertenthal, Eds.). Washington, DC: National Academy Press.

National Research Council. (2007). Taking science to school: Learning and teaching science in grades K–8. (R.A. Duschl, H.A. Schweingruber & A.W. Shouse, Eds.). Washington: The National Academies Press.

Neumann, K., Viering, T., Boone, W.J. & Fischer, H.E. (2012). Towards a learning progression of Energy. Journal of Research in Science Teaching, Nov 2012.

Pavlik Jr., P. I., Maass, J., Rus, V. & Olney, A. M. (2012). Facilitating Co-adaptation of Technology and Education through the Creation of an Open-source Repository of Interoperable Code. In S. A. Cerri, W. J. Clancey, G. Papadourakis & K.-K. Panourgia (Eds.), Proceedings of the 11th International Conference on Intelligent Tutoring Systems (pp. 677-678). Berlin: Springer.

Renkl, A., Worked-out examples: Instructional explanations support learning by self-explanations. Learning and Instruction, 2002. 12: p. 529-556.

Roberts, L.,Wilson, M. & Draney, K. (1997, June).The SEPUP assessment system: An overview. BEAR Report Series, SA-97-1. University of California, Berkeley.

Rus, V., D'Mello, S., Hu, X. & Graesser, A. C. (in press). Recent advances in conversational dialogue systems. AI Magazine.

Schmidt, W. H., Wang, H. C. & McKnight, C. C. (2005). Curriculum coherence: An examination of US mathematics and science content standards from an international perspective. Journal of Curriculum Studies, 37(5), 525–559.

Shute, V. J. (1993). A macroadaptive approach to tutoring. *Journal of Artificial Intelligence in Education,* 4(1), 61-93.

Songer, N.B., Kelcey, B. & Gotwals, A.W. (2009). How and when does complex reasoning occur? Empirically driven development of a learning progression focused on complex reasoning about biodiversity. Journal of Research in Science Teaching, 46, 610–631.

Sottilare, R.A., Brawner, K.W., Goldberg, B.S. & Holden, H.K. (2012). *The Generalized Intelligent Framework for Tutoring (GIFT)*. Orlando, FL: U.S. Army Research Laboratory – Human Research & Engineering Directorate (ARL-HRED).

Steedle, J.T. & Shavelson, R.J. (2009). Supporting Valid Interpretations of Learning Progression Level Diagnoses. Journal of Research in Science Teaching, 46(6), 699-715.

Stevens, S. E., Delgado, C. & Krajcik, J. S. (2009). Developing a hypothetical multi-dimensional learning progression for the nature of matter. *Journal of Research in Science Teaching*.

Valverde, G.A. & Schmidt.W.H. (1997). Refocusing U.S. math and science education. Issues in Science and Technology, 14(2), 60-66.

VanLehn, K. *(*2006*)* The behavior of tutoring systems. International Journal of Artificial Intelligence in Education. 16 (3), 227-265.

VanLehn, K. (2011). The relative effectiveness of human tutoring, intelligent tutoring systems and other tutoring systems. Educational Psychologist, 46, 4, 197-221.

Wilson, M. (2005). Constructing Measures: An Item Response Modeling Approach. Mahwah, NJ: Erlbaum.

Wilson, M. (2009). Measuring Progressions: Assessment structures underlying a learning progression. Journal for Research in Science Teaching, 46(6),716-730.

SECTION IV

FUTURE LEARNER MODELING CONCEPTS

R. Sottilare, Ed.

CHAPTER 17 – Pushing and Pulling Toward Future ITS Learner Modeling Concepts

Robert A. Sottilare
U.S. Army Research Laboratory - Human Research and Engineering Directorate

Introduction

In the previous sections, we examined the learner modeling literature, the state of current practice in learner modeling, and emerging concepts in learner modeling. In this section, we examine ideas related to future capabilities for adaptive tutoring systems, the technologies needed to realize these capabilities, and the maturity of those technologies today. We discuss how breakthrough technologies perceived to be outside the ITS domain today have the potential to change how we think about AI tutors in the future. Recommendations for long-term research are also provided that support both identified needs (technology pull) and innovation (technology push).

A Vision for Future Tutoring Systems

As we look forward toward increasingly intelligent and adaptable tutoring systems, an ontology is needed to characterize essential capabilities and establish standards of performance. Capability definitions might compare and contrast: the degree of tailored instruction that the tutor can provide; how effectively the tutor can support/enable learning; the ability of the tutor to perceive the learner as a basis for tailoring instruction and optimizing performance; or the tutor's compatibility with existing training platforms (e.g., serious computer-based games). Whatever the capabilities of future ITSs, the real measures of success lie beyond learning effect. Future tutoring systems must be easier to develop, access, and use than their counterparts today. They must incorporate reuse standards to reduce time and cost for development. They must provide tailored user interfaces to support usability by learners, domain experts, teachers/instructors/trainers, ITS developers, instructional designers, and researchers.

The theoretical concepts of today will evolve into the practical implementations of tomorrow. The capabilities characterized in ideal future tutoring systems will not just be the result of compromising practicality, mapping (compatibility), and computational complexity (Preface in this book) to realize a workable design, but instead will embody a collaboration between users and tutoring technologies over a lifetime of learning. The seeds of future ITSs are being sown in research that expands the definition of learner models in new directions. A future persistent learner model resides in the cloud, tracks long-term performance, and models competency, values, preferences, goals, and beliefs to help foster trust, creativity, and self esteem within individual learners and teams of learners.

Learner Models in the Future

The four chapters in this section highlight ongoing areas of research that were specifically broken out from the emerging concepts discussed in Section III of this book due to their impact to learning and their anticipated long-term evolution. Each chapter in this section identifies key challenges in developing a more useful and comprehensive learner model, tools and methods for GIFT to build upon. The research discussed points to the importance of comprehensive learner modeling in determining tailored learning experiences over a lifetime.

The chapter by Lester, Mott, Rowe, and Sabourin examines the detection of learner's affective states and their impact on cognition, motivation, and metacognition during game-based tutoring events. The relationship between affect and learning is a critical link in determining optimal instructional strategies and tactics to be employed by the tutor. Significant challenges exist in accurately and unobtrusively determining affect in real-time. Affect detection is further complicated since affect is generally inferred by through observation of the learner by a human or computer-based tutor. Additional challenges arise when a computer must infer affect in complex training environments such as serious games with high degrees of freedom of learner interaction or when a computer must resolve ambiguity between learner behaviors and physiological measures.

The chapter by Burleson and Muldner discusses the future role of ITSs as intelligent creativity supporters. Creativity is a key ingredient for moving learning forward beyond the sum of acquired knowledge and skill. In traditional classroom settings (one teacher and many students), there is generally insufficient time to support a creative curricula tailored to the individual needs of each learner. ITSs have an advantage as one-to-one tutors to provide the special attention needed to foster creativity. Future tutoring systems may be designed to adjust their instruction to promote risk-taking, support learner adaptability, encourage grit, and assess options that lead to creative solutions. A significant challenge may be the ability to efficiently author increased numbers of strategies and content as creativity support tools are linked with adaptive tutoring systems to allow greater flexiblity and assessment of solutions. In other words, ITSs with creative support should be capable of allowing more than a single "right" answer. On the flip side, a considerable advantage might be realized as ITSs are applied to ill-defined domains where creativity is at a premium, and there are multiple serviceable solutions.

The chapter by Regan, Raybourn, and Durlach puts forth a concept for a Personal Assistant for Learning (PAL) that expands the capabilities of ITSs today to include advisory functions (coaching and mentoring). As in GIFT and other tutoring architectures, the PAL learner model will be of central importance to determine instructional strategies for learning, enhance creativity (see Burleson and Muldner, chapter 19 in this book), motivate, activate, and support decision-making. The importance of this chapter in projecting learner modeling capabilities for ITS rests in its illustration of learners as drivers of their own learning experiences as opposed to someone to be guided/led by the tutor. A wealth of information is ready to be mined to enhance our future learner models. Social media is a goldmine of learner preferences, interests, habits, goals, knowledge, and skillsets in time-stamped, context-related bundles of information. These multiple sources of information call for standards for interoperability to support consumption by current and evolving ITS architectures.

The chapter by Fletcher and Sottilare examines how the architectural principles and functions described by GIFT might be extended to support the training of teams. Shared mental models represent team objectives and the actions, both individual and collective, needed to achieve them. These models represent team communication and coordination, team posture, situation, and environment, and team member roles and responsibilities. The focus of this chapter is on shared mental models of cognition and includes models of team purpose, behavior, and functions that are analagous to individual cognitive models in most ITS today. Exploration of team affective models and physiological factors influencing learning are left for future discussions, but an exploration and understanding of shared cognitive models provides insight into how non-cognitive factors might be addressed in the future. Since teamwork differs in the quantity and quality of communication and coordination required, major challenges rest in how to measure how good communication, coordination, and other factors (e.g., goals, roles, individual knowledge and skills, and preferences) support (or detract from) optimal team performance.

The Long View of GIFT

The contributors to this section of the book offered recommendations for developing the learner model component of GIFT across different dimensions (affect modeling, creativity support, data mining and open learner modeling, and shared mental models). The recommendations addressed substantial challenges and opportunities that are envisioned to evolve over an extended period of time due to their complexity. The following enumerates recommended actions for consideration in the long-term view of GIFT. Some of these recommended actions are already defined with known value (technology pull) and some are more speculative (technology push) in that their impact is difficult to predict at this time.

1. Significant effort has been expended to develop affect detectors for individual learners. A systematic analysis based on empirical studies should be conducted to evolve standards that can be applied across training tasks and are suitable for both individual learners and teams of learners.

2. Advance a new class of learning technologies focused specifically on creativity support tools that are linked directly to tutoring architecture components for domain knowledge and motivation. A systematic analysis based on empirical studies should be conducted to evaluate the effectiveness of these tools in teaching creativity and building innovation skills in individual learners and teams of learner.

3. Develop open learner models to support intelligent selection of learner control during tutoring sessions. A systematic analysis based on empirical studies should be conducted to develop and evaluate open learner models to optimize learning.

4. Enhance data-mining techniques to support persistent learner models that are automatically updated over time to reflect changes in preferences, interests, goals, knowledge, and skills. A systematic analysis based on empirical studies should be conducted to determine the effect size of various learner traits on cognition and motivation.

5. Enhance data analysis techniques to support rapid development of expert and misconception models based on crowd sourcing. Develop standard tools and methods to allow for plug and play expert and misconception models in standard ITS architectures like GIFT.

6. A prototype has been developed that implements characteristics of GIFT, including the learner model. A systematic analysis based on empirical studies should be conducted to evolve a similar comprehensive model for teams. These shared mental models should be applicable across various task domains including cognitive (e.g., problem solving), affective (e.g., value judgments), and psychomotor (e.g., controlled movement of the body) tasks. A library or repository should be established to house these models and support standards for instructional design of team training.

7. Develop a formal ontology for GIFT with the support of the ITS community to help focus attention on critical missing learner modeling elements to support authoring, instructional management, and analysis constructs. Extend this ontology to incorporate other aspects of ITS research over time.

8. Develop standards to classify tutoring system capabilities in critical areas (e.g., learning effect size, accuracy of cognitive and affective state classifiers) per the adaptive tutoring learning effect chain (see Preface of this book for individual tutoring or Fletcher and Sottilare, chapter 22 of this book for team tutoring).

9. For future capabilities to support adaptive tutoring of individuals and teams, tools and methods are needed to deeply engage learners and support their deep learning; encourage adaptability, grit (tenacity), and innovation in seeking solutions in military domains with multiple solutions; appropriate control by the learner in their own learning experiences; and shared mental models for teams. A stringent and extensive set design principles should be developed to support enhanced tutor-learner interaction and higher learning effect. A systematic analysis based on empirical studies should be conducted to develop and evaluate the following functions:

   - *Sensors* – behavioral and physiological sensor design should be unobtrusive so as not to interfere with or distract from learning processes during tutoring sessions.

   - *State classifiers* – learner modeling processes should be very accurate (near 100%) in using learner data to determine the cognitive, affective, and physical states (e.g., confusion, frustration, engaged concentration, boredom, fatigue), which are most influential with respect to readiness-to-learn and the learning gains of both individuals and teams.

   - *Measures of success* – tutor assessment engines should be designed to support easy authoring and linkage of success metrics to knowledge and skill acquisition, and performance,

   - *Adaptive instruction and support* – knowledge of the learner's states and traits should lead to optimal selection of instructional content, strategies, and tactics along with additional adaptation based on the learner's questions and goals,

   - *Individual and team modeling* – information about individual learners and teams of learners should represent their individual/collective domain competency, motivation, and expectations of success based on competency and task complexity (e.g., cognitive problem solving, affective valuing),

CHAPTER 18 – Learner Modeling to Predict Real-Time Affect in Serious Games

James Lester, Bradford Mott, Jonathan Rowe, and Jennifer Sabourin
North Carolina State University - Department of Computer Science

Introduction

In recent decades research on human emotion has revealed that affect plays a central role in human cognition. Historically, emotion was viewed as separate from cognition, and only recently has there been consensus that affect is a central component of rational behavior and social interaction (Forgas, Wyland & Lahan, 2006). These findings have motivated the goal of designing affect-sensitive computer systems capable of recognizing and responding to users' emotional states. This has led to the development of affective systems with a wide variety of capabilities and purposes, such as therapeutic virtual agents, mood detection of large social network groups, and empathetic tutorial systems (Picard, 1997). Applications such as these require that three critical decisions must be made to inform how affect will be modeled and detected: (1) How will affect be represented? (2) What signals will be used for affect detection? (3) How will affect predictions be used?

Devising frameworks for representing affect has been the subject of considerable study in psychology and cognitive science (Ortony, Clore & Collins, 1990; Russell, 2003). Psychologists have proposed a broad range of solutions for representing affect. These representations have left computational researchers with a variety of frameworks from which to choose. For example, affect may be viewed as a concrete categorical state, such as *anger* or *happiness*. Alternatively, affect can be represented among a variety of dimensions such as valence or arousal, where a state is only differentiated by its position on the scale. Further, affect can be seen not as a state itself but as an emergent phenomena consisting of action tendencies, physiological responses, and conditional precedents. Each of these representations introduces benefits and challenges when used in computational systems, and these tradeoffs must be considered within the contexts of specific applications. This chapter explores the role of computational models of affect in serious games, with a focus on real-time affect prediction.

Unrealized Capabilities and Gaps

Computational Models of Affect

Affect is fundamentally a hidden state. Detecting affect, when performed by either humans or computers, requires inference from observable signals. Selecting signals to incorporate into an affect-detection system requires consideration of several factors such as cost, invasiveness, and predictive value (Calvo & D'Mello, 2010). Common signals include physiological information such as heart rate and skin conductance, which can be measured through physiological sensors. Expressive traits such as facial expressions, posture, gaze, and gesture are other informative channels that can be measured with web cams or specialized sensors. Behavioral evidence of affective states is a complementary modality, and it is often inferred from logs of user interactions with software. Further, typed statements or spoken sentiment can be analyzed in a variety of meaningful ways to infer emotion. Each of these channels can be considered independently, but many successful approaches to modeling combine input from multiple modalities to arrive at a more complete and accurate representation of the states they are endeavoring to model.

A primary concern that often guides investigations of affect is the application for which the inferred knowledge about affective state will be used. In some cases, it is useful to simply know how the user is feeling as an evaluation of the system itself. In other cases, this information may be incorporated into run-time interactions to dynamically improve user experiences or other important outcomes. Differences in objectives may lead researchers to consider multi-dimensional approaches, which are particularly beneficial if a single trait is being monitored over time. However categorical definitions are better for communicating with the user about their state. Similarly, if the application is multi-platform and intended to be used by individuals in a broad range of settings, some physiological sensors may be impractical. However, if the objective is inferring highly accurate representations of states, incorporating as many signals as available may be the best route. In many ways, the final application of the affective information drives all other decisions.

Modeling Affect in Serious Games

An important application of affective modeling capabilities is in serious games. Serious games combine two affectively charged activities, learning and gameplay, making them an interesting avenue for exploring issues of affective modeling. Affect is a critical component of learning and has been shown to influence how students process information, approach learning activities, and feel about themselves and their abilities (Baker, D'Mello, Rodrigo & Graesser, 2010; Pekrun, Goetz, Titz & Perry, 2002; Picard, et al., 2004). Meanwhile, though enjoyment and happiness are viewed as fundamental components of play, games can introduce degrees of frustration, confusion, sadness, and even anger. Together the rich affective experiences associated with learning and gameplay motivate the need to explore affective modeling in serious games.

Serious games have considerable capacity to evoke a broad range of emotions, because they can contextualize learning and training activities within meaningful contexts that approximate authentic settings. For example, games often incorporate narrative plots, which in many ways are fundamentally emotion-inducing: interesting stories are defined by conflict, uncertainty, suspense, and surprise. Furthermore, they are often host to interactions with believable virtual characters, fantasy or simulated settings, and dynamically unfolding situations. These types of learning environments lend themselves to a broad range of emotional responses that are likely to shape learning processes, but may be atypical of alternate educational environments that deliberately separate educational content and application context.

For a range of educational settings, evoking these types of emotional processes during learning may be desirable, or even essential, to produce learners who will be capable of performing skills in realistic settings that will similarly evoke these emotions. However, creating educational environments that can evoke these types of emotions raises a number of issues:

1. How can we recognize students' emotional states, including emotions that may not be commonly observed in non-game settings, such as fear, surprise, disorientation, anxiety, excitement, curiosity, and sadness, so we that we can better understand which, and how, emotions impact student learning?

2. How, and to what extent, can we effectively model students' emotions of these types in real time, including predictive models that converge on accurate predictions of student emotions in advance of their occurrence based on expected narrative states and affect-sensitive learner models?

3. How can we devise models that dynamically tailor events in serious games, including tutorial events, narrative events, and game parameters, to aid students in self-regulating their learning and affective processes?

Recent work in this area has explored many varied components of affect in serious games. For example, Conati et al. (2009) have explored probabilistic models for affect recognition in *Prime Climb*, a serious game for young children learning number factorization. These models incorporate in-game context data along with physiological sensors to arrive at accurate predictions of student emotion. *FearNot!* is a serious game for teaching anti-bullying behavior to school children (Paiva, Dias, Sobral & Aylett, 2004). This environment seeks to model the affective states of virtual characters to drive empathetic relationships with the learner. Robust emotion models have also been incorporated into the *BiLAT* serious game (Kim, et al., 2009). This environment seeks to teach military personnel skills for multicultural negotiation and uses affective information to drive character interactions and responses.

This work has highlighted the role that affect can play in the development and understanding of serious games and points to areas where future work is needed. First, much work in modeling user affect takes place in highly structured environments. In these systems, users typically have a small set of available actions and have a clear indicator of correctness and progress. This simplifies the identification of features relevant to affect modeling. However, it will be important to begin exploring how affect can be modeled in more open-ended exploratory games. There may be different sets of affective phenomena at play in these environments and it is likely that a different set of tools and techniques will be needed to build successful predictive models in these systems. Another important area of future work is more exploration of how affect models can be incorporated back into serious games. While FearNot! and BiLAT use affective information to drive narrative and character interactions, there is less exploration on how this information can be used to reason about the learner or guide tutorial strategies. A deeper understanding of both sides of this affective picture is needed to fully describe the important role affect plays during interactions with serious games.

Future Concept

Given these gaps in the research on computational models of affect, we envision three major thrusts of research for real-time affect detection in serious games. First, we anticipate the potential for sizable advances in early prediction of student emotions during game-based learning. Second, we anticipate that data-driven models of student affect will be combined with theoretical frameworks of student learning and emotion to improve models for assessing student knowledge acquisition and transfer. Third, we anticipate that real-time models of student affect will be incorporated into tutorial systems capable of personalizing events in serious games to scaffold student learning and promote sustained engagement.

Predicting Student Affect in Guided Inquiry-Based Serious Games

Modeling student affect in serious games poses distinct challenges relative to other educational systems. These challenges primarily stem from two components of these systems: (1) user actions and goals often unfold in a multitude of different orders within rich simulated environments, and (2) models must meet the run-time performance requirements of games. The open-ended nature of many serious games means that learners are free, and encouraged, to approach the task in any way they choose. This makes selecting contextual features for modeling tasks significantly challenging. Furthermore, most contemporary theories of emotion suggest that affect is often generated in response to the success or failure of one’s goals, and how this success or failure came about (Elliot & Pekrun, 2007; Ortony et al., 1990). In serious games, students’ goals may be unclear and without this knowledge, it will be difficult for the system to reason about success or failure and resulting affective states.

Recent work has investigated predictive models of students’ educational goals and plans in serious games. These models consider sequences of student actions in a game environment, and provide early predictions about the goal a student seeks to next achieve or the problem-solving plan currently being executed. Models of student intentions hold considerable promise for informing predictions of students' affective states. Affective experiences are often a direct result of an individual's intentions and how these are or are not being realized in the world. An accurate prediction of a student's goals or intentions enables a system to make inferences on whether these goals are being achieved and offer the opportunity to reason about attribution of success or failure. Together, intention and attribution offer significant insight into an individual's affective state (Ortony, Clore & Collins, 1990) and are an important component of a context-based affect recognition model.

State-of-the-art serious games often employ computationally intensive graphics rendering and simulation capabilities. These technologies require substantial computational resources, and therefore limit the resources available for other system components such as real-time affect prediction models. While many successful models of affect prediction have been achieved by combining multiple data streams, it will be important to identify the data sources that provide the most benefit while imposing the least amount of computational overhead. For example, rich data from physiological sensors may be highly informative, but unless it can be effectively processed along with games' other required computational requirements, it may not be useful in practice. This highlights the importance of developing algorithms and data streams that are able to use computational resources as they become available and make predictions efficiently in real time.

Affect-Informed Models of Student Learning

While affect predictions alone can provide meaningful information about student engagement while interacting with a serious game, another promising direction is identifying how affect predictions can be incorporated back into the understanding of the user's learning. Learner models are used to guide interactions and understanding of the student in a variety of systems. These models first seek to assess student knowledge, and then adapt tutorial interactions accordingly to maximize learning gains. A student model that incorporates affect may provide a richer picture of student learning. For example, affect has been shown to influence whether students are more likely process information in a bottom-up or top-down fashion (Pekrun et al., 2002). By considering a student's affective state as part of the learner model, considerations can be made about how a student is processing information, the connections that are likely being made, and how further information should be presented.

Incorporating affective channels into models of student knowledge acquisition and transfer for serious games hold considerable promise, particularly in domains where real-world applications of the learned knowledge are affect-intensive. Affect has been shown to influence cognition in many ways, and exploration of how these relationships can be used along with students' affective states is an important direction for developing comprehensive learner models.

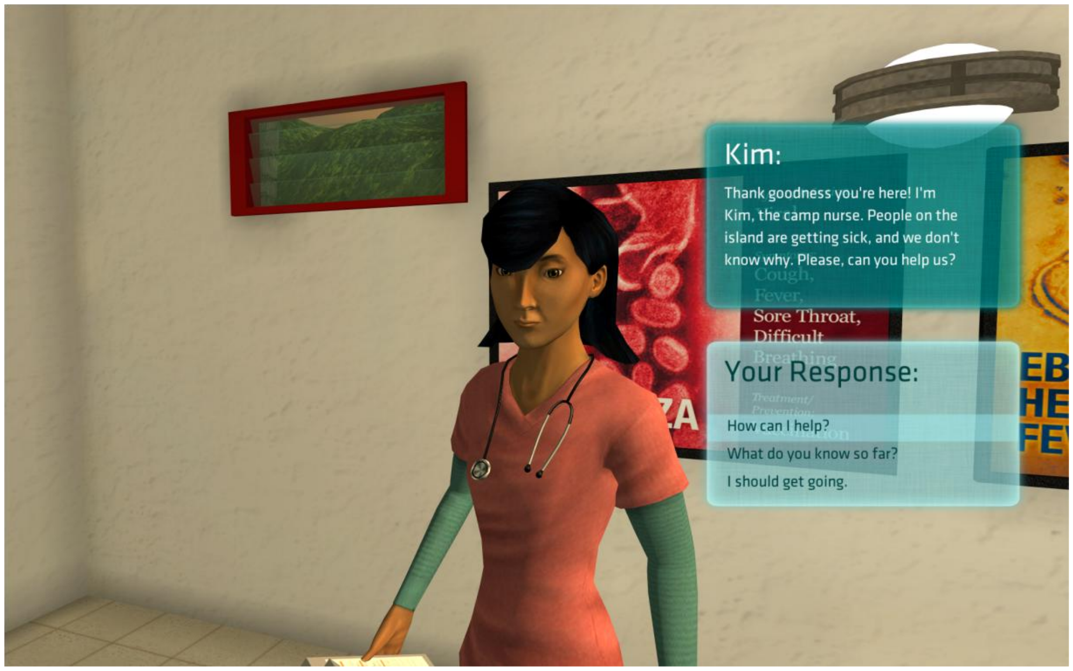
Affect-Informed Tutorial Planning in Serious Games

Real-time models of students' affective processes could inform models for dynamically enhancing students' learning experiences in serious games. By combining serious games and ITSs, it is possible to devise models that tailor game mechanics, missions, rewards, and difficulty levels to scaffold student learning. In the case of story-based games, narrative-centered tutorial planners are a form of integrated pedagogical planner and interactive narrative director agent that discreetly support students' learning processes by tailoring story events. Narrative-centered tutorial planners consider the state of the student, interactive narrative, and learning progression to make decisions about how to adapt event sequences in the storyworld to support students' learning, problem solving, and engagement. For example, a narrative-centered tutorial planner may modify a virtual characters' behavior to provide additional hints and explanations to a student requiring special assistance. Alternatively, the planner could introduce novel goals and sub-plots to provide remediation for students, or opportunities for assessing advanced skills.

Real-time models of students' emotional states are well positioned to enhance the capabilities of these narrative-centered tutorial planners. However, there has been little work to date investigating how real-time affect models can enhance narrative-centered tutorial planners' effectiveness. By making accurate and early predictions of students' emotions, narrative-centered tutorial planners could direct characters to provide personalized affective assistance, such as encouragement when a student is feeling confused, empathy when a student is feeling anxious, or advice to take a break when a student has experienced prolonged frustration while investigating a complex problem-solving task. These types of capabilities are poised to significantly enhance students' cognitive-affective processes in story-based serious games.

Illustrative Scenario

In order to illustrate these future concepts about real-time affect modeling in serious games, we describe a vision for affect-driven models in a game-based learning environment for middle school microbiology, CRYSTAL ISLAND (Rowe, Shores, Mott & Lester, 2011). CRYSTAL ISLAND (Figure 18-1) features a science mystery, in which the student has arrived on a remote island to discover that the research team that has been established there has fallen ill. The camp nurse explains that they have not been able to identify the cause or type of illness and asks for the student's help. The student then works to collect clues by talking with virtual characters, running tests on objects in the world, and reading related books and posters. Once the student arrives at the correct source and type of illness and proposes a diagnosis, the student has solved the mystery and completed the game.



Thank goodness you're here! I'm Kim, the camp nurse. People on the island are getting sick, and we don't know why. Please, can you help us?

How can I help?
What do you know so far?
I should get going.

Figure 18-1. Screenshot of the CRYSTAL ISLAND game-based learning environment

As an open-ended game-based learning environment, CRYSTAL ISLAND features many goals and objectives that students may be working towards at any given time. For example, one student may be actively trying to identify the common symptoms among patients, while another may be trying to identify common food items that may be the source of the illness. Distinguishing between these two goals is critical for identifying whether a student is being successful at their goals and how they may feel as a consequence. If the student is trying to identify common food sources, but only hears about symptoms, they will likely be frustrated, while the other student would be feeling confident and hopeful that they will progress successfully.

Successful affect recognition in CRYSTAL ISLAND introduces opportunities for affect-driven learner models. Affective information can provide insight into how a student is learning, and the activities they are likely to pursue. For example, a student who is feeling confused is likely missing some piece of information that is critical to their understanding. A learner model that is able to assess knowledge and affect may be able to identify these gaps in understanding and drive tutorial planning so that the student is able to overcome this cognitive dissonance.

Another possibility for incorporating affect in tutorial and narrative planning revolves around the system's ability to foster positive affective states and engagement. For example, if the system detects that a student is frustrated or bored with the learning task, it may be advantageous to allow the student some respite from the difficult material. Game-based learning environments like CRYSTAL ISLAND allow opportunity for students to be engaged in the environment but not on the learning task specifically. Students may interact with the game environment's physics simulation by jumping on and stacking objects, or by simply exploring the rich 3-D world. These types of actions may have positive affective benefits that can be encouraged if the system recognizes the need for affective regulation.

Discussion

The opportunities that we have outlined for real-time affect detection in serious games highlight a number of implications for research and design of intelligent tutoring systems and related learning technologies. In particular, the key capabilities and tools provided by the GIFT framework are synergistic with the requirements and opportunities that we have discussed for modeling student affect. GIFT provides three primary functions: authoring capabilities for constructing novel learning technologies, instruction that integrates tutorial principles and strategies, and support for evaluating novel educational tools and frameworks. These capabilities provide a foundation for investigating real-time affect modeling in serious games, and further extensions to GIFT will enable studies of generalizable affect modeling frameworks.

Affect in Serious Games

Creating serious games can pose significant costs, due to the need for aesthetic 3-D assets, novel interactive narratives, robust simulation models, and believable virtual agents, all meeting the run-time performance requirements of games. While serious games often do not compete directly with commercial entertainment-focused games, the continually rising bar for games' production values also increases expectations of serious games' complexity and aesthetics. Creating serious games requires close collaboration between interdisciplinary teams of digital artists, computer scientists, subject matter experts, game designers, and instructors. Consequently, devising tools that can reduce the authoring costs associated with integrating adaptive tutoring capabilities for real-time affect modeling and scaffolding will represent a substantial advance. Further support for reducing authoring costs and increasing component reusability through GIFT, as well as research demonstrating the authoring benefits of these technologies, would be valuable to serious game developers.

Many serious games take advantage of advanced 3-D graphics and real-time agent behavior algorithms to create immersive, believable virtual environments. However, these characteristics are computationally intensive, and typically must be performed at more than 30 frames per second to preserve visual fidelity. In many cases, this leaves limited local resources for updating computationally sophisticated models of student emotion or learning processes. Offloading these modeling and reasoning capabilities to external modules, especially modules hosted on external servers, is a promising approach for supporting computationally intensive probabilistic models of students' learning and affective processes, while meeting the run-time performance requirements of serious games that run on students' local computing hardware. GIFT's modular framework and service-oriented architecture are conducive to decoupling learner modeling and pedagogical planning decisions from students' client machines.

GIFT also provides broad support for external sensors, which can supply real-time data on students' physiological state such as skin conductance, heart rate, posture, and eye gaze. Self-reports provide a useful window into students' affective processes, but they may be disruptive if presented during gameplay, and have limited accuracy particularly in cases where they are provided after an experience has concluded, and thus temporally removed from the actual occurrence of the emotion. Physiological sensors in many cases may be able to provide complementary information about students' affective states without disrupting gameplay and learning experiences. Furthermore, sensor data are objective, which removes some of the limitations in depending on students' abilities to recognize their own emotions and effectively and precisely report them.

Affect and the GIFT Architecture

The GIFT framework encompasses a modular architecture, which includes the following tutor components: a sensor module, a learner module, a pedagogical module and a domain module (Sottilare, Goldberg, Brawner & Holden, 2012). Given this architecture, there are a number of questions and opportunities concerning how to effectively incorporate affect sensitivity within the GIFT framework. Future extensions to GIFT in service of real-time affect modeling will likely touch upon each of these four modules.

Each tutor module has its own distinct opportunities for processing affective information and communicating this to other modules. The sensor module is responsible for collecting and synthesizing information from various sources, such as webcams, electrodermal activity sensors, and pressure-sensitive mice. This module must handle raw multimodal input from multiple concurrent sensors and provide output metrics that are useful for modeling the affective states of users. For example, a webcam capturing a large stream of video data could be analyzed to identify facial expressions, posture, or non-verbal gestures indicative of affective states. In many cases, these sensors can produce large quantities of raw data. Consequently, as additional sensors are incorporated into GIFT, the sensor module must be capable of handling the memory and processing demands imposed by these new types of data, which are distributed across the collection, storage, cleaning, and transformation stages of data management.

The learner module is responsible for representing the cognitive and affective states of learners using processed sensor data as well as learner performance data from the environment. Affective representations can include a variety of features, such as emotional state, self-efficacy, motivation, interest, and intention. Accurately modeling each of these components may require detailed knowledge of the task and learning environment in addition to inputs provided by the sensor module. For example, modern appraisal-based theories of emotion depend heavily on how learners' actions and intentions play out in particular environments. Consequently, learner modules that implement these theories need to have access to information about the learning environment. Furthermore, affective and cognitive states are highly intertwined and must be considered in concert. For example, a learner's performance and knowledge may influence emotional state and self-efficacy, which will then impact how the learner approaches the task moving forward.

The pedagogical module receives information about learners' current and predicted states, and it uses this to guide instructional strategies. One challenge that must be addressed by this module is balancing the tutoring system's affective and cognitive goals. For example, a more difficult learning task may be beneficial for increasing knowledge, yet an easier task may increase confidence and enjoyment. Furthermore, this module must consider a variety of strategies to bring about cognitive and affective improvements. Hints and feedback are commonly delivered to guide students' knowledge acquisition and problem solving; however, these strategies may also include affective content. For example, empathetic feedback may encourage students to continue feeling positively in spite of poor performance. Alternatively, hints on effective emotion regulation strategies could be provided to students who appear to be struggling. Tailored support of learners' cognitive-affective states is likely to have substantial positive impacts on increasing time-on-task and sustaining learner engagement.

The domain module contains information about both the content area and the task environment in which learning interactions take place. It includes explicit representations of the types of feedback and adaptation capabilities supported by the learning environment, which can be used to enact the proposed strategies suggested by the pedagogical module. Such adaptations may involve virtual agents capable of verbal and non-verbal affective expression, tailored events in the learning environment's narrative, or dynamic adjustments to task difficulty. This module is also responsible for assessing student performance and identifying learner behaviors that are indicative of cognitive-affective states. For example, student off-task or gaming behaviors may serve as powerful indicators of student engagement and motivation.

Devising computational models for real-time affect prediction in serious games offers significant promise. Initial versions of GIFT have implemented modules primarily devised to effectively scaffold student knowledge acquisition and skill mastery. Future efforts to endow GIFT with affect modeling capabilities, such as the features described above, will require additional research and development efforts to expand the capabilities of each tutor module.

Recommendations and Future Research

GIFT provides a technological foundation for investigating predictive models of affect that can be generalized across different serious games. There are several promising directions for extending the capabilities of GIFT's modules to integrate comprehensive support for recognizing, understanding, and expressing affect in support of learning. Specifically, achieving the vision we have outlined for real-time affect detection in serious games through GIFT will require solutions to several research questions:

1. How can generalizable affect recognition, understanding, and expression capabilities be implemented within each of the modules of tutoring architectures such as GIFT?

2. How should interfaces between tutoring modules be developed to ensure robustness in handling different configurations of affect sensors and training environment capabilities?

3. How can generalizable, modular implementations of predictive affect models operate within the run-time performance requirements of serious games?

4. How can game-specific models for affect support be incorporated into generalizable tutoring architectures and transfer to alternate tasks and domains?

These questions highlight the challenges associated with investigating generalizable intelligent tutoring architectures capable of real-time affect modeling in serious games. With regard to GIFT specifically, it will be important to identify the key issues, and solutions, for incorporating affect sensitivity within each of the tutor modules in the GIFT architecture. When designing these components, one must consider how these components communicate with one another, and how the system should be configured to support cases where components are missing. For example, physiological sensors are highly beneficial for affect recognition, but may not be available in all cases. Consequently, a learner model relying on output from such a sensor would need to be adapted, or gracefully deactivated, in a manner that minimizes negative impacts on other modules. Similarly, different genres of serious games have distinct capabilities and affordances. For example, serious games with believable virtual agents or rich narrative contexts may present different opportunities for affective feedback than serious games without these features. In cases such as these, pedagogical modules that recommend empathetic character behaviors or story event adaptations to provide affective support require mechanisms for handling cases where learning environments do not support these types of intervention naturally.

Additional challenges are raised by the computational demands of serious games. Games impose significant run-time performance requirements because they are computationally intensive, and they must balance graphics rendering and simulation capabilities alongside affective and tutorial modeling. Predictive affect models must balance efficiency, accuracy, and relevance to shape effective pedagogical interventions embedded within game environments. Finally, it will be important to investigate the ways in which findings related to the benefits and constraints of supporting affect in serious games extends to other tutoring environments. Advancements such as these would likely reduce the costs of devising new intelligent tutoring systems, as well as enhance their effectiveness for different learning settings.

References

Baker, R. S., D’Mello, S., Rodrigo, S. K. & Graesser, A. C. (2010). Better to Be Frustrated than Bored: The Incidence, Persistence, and Impact of Learners’ Cognitive-Affective States during Interactions with Three Different Computer-Based Learning Environments. *International Journal of Human-Computer Studies*, *68*(4), 223–241.

Calvo, R. A. & D’Mello, S. (2010). Affect Detection: An Interdisciplinary Review of Models, Methods, and Their Applications. *IEEE Transactions on Affective Computing*, *1*(1), 18–37.

Conati, C. & Maclaren, H. (2009). Empirically building and evaluating a probabilistic model of user affect. *User Modeling and User-Adapted Interaction*, *19*(3), 267–303.

Elliot, A. J. & Pekrun, R. (2007). Emotion in the Hierarchical Model of Approach-Avoidance Achievement Motivation. In P. A. Schutz & R. Pekrun (Eds.), *Emotion in Education* (pp. 57–74). London: Elsevier.

Forgas, J., Wyland, C. & Lahan, S. (2006). Hearts and minds: An introduction to the role of affect in social cognition and behavior. In J. P. Forgas (Ed.), *Affect in Social Thinking and Behavior* (pp. 3–18). New York: Psychology Press.

Kim, J., Hill, R. W., Durlach, P. J., Lane, H. C., Forbell, E., Core, M., Marsella, S. C. & et al. (2009). BiLAT: A game-based environment for practicing negotiation in a cultural context. *International Journal of Artificial Intelligence in Education*, *19*(3), 289–308.

Ortony, A., Clore, G. & Collins, A. (1990). *The cognitive structure of emotions*. Cambridge University Press.

Paiva, A., Dias, J., Sobral, D. & Aylett, R. (2004). Caring for agents and agents that care: Building empathic relations with synthetic agents. *Proceedings of the 3rd International Joint Conference on Autonomous Agents and Multiagent Systems* (pp. 194–201).

Pekrun, R., Goetz, T., Titz, W. & Perry, R. (2002). Academic Emotions in Students’ Self-Regulated Learning and Achievement: A Program of Qualitative and Quantitative Research. *Educational Psychologist*, *37*(2), 91–105.

Picard, R. (1997). *Affective Computing*. Boston: MIT Press.

Picard, R. W., Papert, S., Bender, W., Blumberg, B., Breazeal, C., Cavallo, D., Machover, T., et al. (2004). Affective Learning — A Manifesto. *BT Technology Journal*, *22*(4), 253–269.

Rowe, J. P., Shores, L. R., Mott, B. W. & Lester, J. C. (2011). Integrating learning, problem solving, and engagement in narrative-centered learning environments. *International Journal of Artificial Intelligence in Education*, *21*(2), 115–133.

Russell, J. (2003). Core affect and the psychological construction of emotion. *Psychological Review*, *110*, 145–172.

Sottilare, R. A., Goldberg, B. S., Brawner, K. W. & Holden, H. K. (2012). A modular framework to support the authoring and assessment of adaptive computer-based tutoring systems (CBTS). *Proceedings of the Interservice/Industry Training, Simulation, and Education Conference*.

CHAPTER 19 –Intelligent Creativity Support

Winslow Burleson and Kasia Muldner
Arizona State University - School of Computing Informatics and Decision Systems Engineering

Introduction

We present an argument for the advancement of Intelligent Creativity Support (ICS) systems as an integrating framework for ITSs, affective computing, and creativity support tools, in a manner that closely aligns each of these technologies and research agendas with the componential model of creativity, i.e., domain-relevant expertise, intrinsic motivation, and creative thinking style. We also present strategies for developing and evaluating student models for the just-in-time assessment of creativity.

While there are over one hundred definitions of creativity (Amabile, Barsade, Mueller & Staw, 2005), there is consensus that it entails a product, idea, or process that is novel and useful (Mayer, 1999). Creativity is at the core of all societal advancements. However, it is also present "not only when great historical works are born but also whenever a person imagines, combines, alters, and creates something new, no matter how small" (Vygotsky, 2004).

Creativity has been described as the most vital economic resource of our time (Florida, 2002; Kaufman & Beghetto, 2009) and the U.S. Council on Competitiveness has indicated that it will be the top factor determining America's success in the 21st century (Robbins & Kegley, 2010; Wince-Smith, 2006). Thus, understanding how to foster creativity skills is a crucial societal goal (Tripathi & Burleson, 2012). U.S. universities, colleges, and K–12 school systems can play a fundamental role in producing an innovative and creative workforce, by helping students develop such skills (Robbins & Kegley, 2010; Vance, 2007; Wince-Smith, 2006). Indeed, the 21st Century Skills initiative (Trilling & Fadel, 2009) and Common Core Standards (NGA & CCSSO, 2012) call for teaching creativity, innovation, and deep problem-solving abilities.

Unfortunately, various challenges have hindered the adoption of creativity instruction and practices in traditional classrooms (McCorkle, Payan, Reardon & Kling, 2007). For one, few teachers have been trained in how to teach creativity (Mack, 1987). More importantly, classroom settings do not enable teachers to provide the individualized support needed for effective creativity facilitation. In particular, while personalized instruction has tremendous potential to improve student learning (Cohen, Kulik & Kulik, 1982; Lepper, 1988), affect (motivation and emotion) (Lepper, 1988; Picard, 1997), and metacognitive skills (Bielaczyc, Pirolli & Brown, 1995), providing a human tutor for each student is simply not practical. Given these challenges, most of the work thus far reflects anecdotal, descriptive data (Ma, 2006; Robbins & Kegley, 2010; Runco, 2004; Scott, Leritz & Mumford, 2004), although some exceptions exist (Cheung, Roskams & Fisher, 2006; Clapham, 1997; Dewett & Gruys, 2007).

Since ITSs can provide large-scale instruction that continuously adapts to learners' needs (Aleven, McLaren, Roll & Koedinger, 2006; Arroyo, Cooper, Burleson, Muldner & Christopherson, 2009; Koedinger, Anderson, Hadley & Mark, 1997; Self, 1998; VanLehn et al., 2005), they present a unique opportunity to address issues associated with teaching creativity. ITSs have already successfully improved domain learning by tracking students' problem-solving progress, providing tailored help and feedback, and selecting appropriate problems (Shute & Psotka, 1996; VanLehn et al., 2005). However, ITS have also been criticized for over-constraining student problem solving and over-emphasizing shallow procedural knowledge, and therefore not properly addressing 21st century higher-order skills like critical thinking and creativity (Trilling & Fadel, 2009).

We present strategies for designing a ICS system to foster student creativity during Science, Technology, Engineering and Mathematics (STEM) activities. The ICS framework is situated within Amabile's validated and broadly adopted componential model of creativity (1983). Amabile's model highlights three factors within an individual that are needed for creativity: domain knowledge, motivation, and creative thinking styles. Moreover, Amabile and others have demonstrated that positive affect contributes to creative problem solving (Isen, 2004; Isen, Daubman & Nowicki, 1987), leading to increased intrinsic motivation, deeper exploration, and more appropriate outcomes or solutions. Our goal is to have ICS integrate and leverage traditionally isolated technological components that are critical to advancing a student's creative capacity (Figure 19-1): (1) domain relevant knowledge supported by ITS; (2) affect (motivation and emotion) fostered by Affective Learning Companions (ALCs); and (3) creative thinking skills scaffolded by Creativity Support Tools (CSTs). The ICS design can also implicitly account for external factors that influence creativity, such as evaluation and time pressure (Amabile, 1983; Amabile et al., 2005). The ultimate goal of the ICS strategy is to extend traditional ITS instruction with personalized affective support and metacognitive creativity training to improve creativity and learning outcomes.

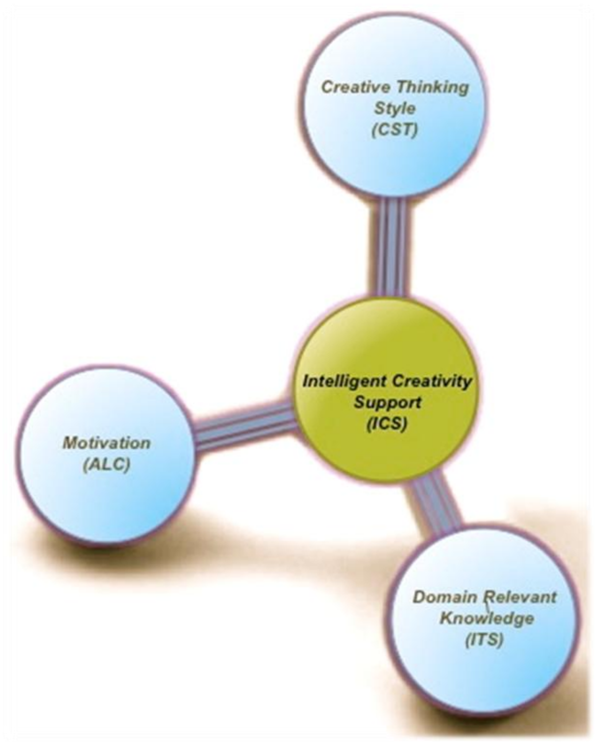

Creative Thinking Style (CST)
Intelligent Creativity Support (ICS)
Motivation (ALC)
Domain Relevant Knowledge (ITS)

Figure 19-1. Advancing a new class of cyberlearning technologies, ICS will integrate personalized support with Amabile's componential model of creativity. ICS will combine ITSs to increase domain relevant knowledge; ALCs to foster motivation; and CSTs to advance creative thinking styles.

Student Models for Just-In-Time Assesment

To provide creativity support tailored to a given student's needs, an ICS requires a *student model* (VanLehn, 1988) that assesses students' attributes relevant to creativity processes and outcomes throughout their educational activities. To prepare to conduct this research, we have taken steps in this direction through related work searches that have highlighted a preliminary set of attributes that we will take into account and extend as needed. These attributes are encapsulated by Amabile's componential model of creativity and related research as follows:

Domain-Relevant Knowledge. Amabile (1983) shows that the more one knows, the more opportunities there are for creativity, something the ICS student model needs to account for in its assessment of a student's creativity. Also related to the assessment of creativity is the fact that its very definition involves the production of a novel idea or problem-solving step – the most natural way for a model to determine novelty is whether the student already possessed the knowledge related to the idea or step or if it was constructed on the spot.

Affect (Motivation and Emotion). How students feel greatly influences the creativity process and its outcomes. Thus, the ICS model will rely on the data from affective sensing devices as well as tutor variables to assess states like intrinsic motivation, central to Amabile's theory (Amabile, 1983), as well as other affective states such as frustration (e.g., indicating Stuck!) and flow.

Metacognition Related to Creative Thinking Styles. The third element of Amabile's theory pertains to what she terms as "creative thinking style," such as how flexible and imaginative people are in their approach to problems, indicating the metacognitive skills required for creativity.

The ICS creativity student model will represent and infer information related to these three attributes. For the modeling of knowledge and metacognition, we will build student models via established techniques (e.g., Conati, Gertner & VanLehn, 2002; Corbett, McLaughlin & Scarpinatto, 2000; Mitrovic, 2012; Reye, 2004) for modeling of these attributes. Specifically, we will use cognitive and metacognitive task analysis to identify fine-grained skills needed to solve a problem (*knowledge*) and for creativity in general (*metacognition*, including, for instance, divergent thinking). These skills can be computationally represented using a *rule-based* approach that enables the system to automatically model both the target solutions and skills sets (Anderson, 1993). This is accomplished by tying parameters to each rule to represent the probability that the student knows the corresponding skill, which "fire" when a certain threshold is exceeded. In addition, this approach can be used to provide the backbone of a Bayesian network that makes the structure of the student knowledge and metacognitive skills explicit, as in (Conati et al., 2002). Overall, this probabilistic approach has the advantage of recognizing that modeling student knowledge and metacognition is not a black and white process, since there is typically inherent uncertainty arising from, for instance, student slips and guesses (Reye, 2004) and/or lack of direct evidence on student state of interest (e.g., divergent thinking).

For the modeling of affect, initially, we will refine our existing student models developed in our work (e.g., Arroyo et al., 2009) and use their output as inputs to the ICS creativity model. These models already capture attributes that are relevant to the research at hand (e.g., interest, related to intrinsic motivation and Flow, frustration) by relying on data from the sensing devices and tutor variables. However, as mentioned above, these models do not take into account the uncertainty inherent in assessing affect as other existing affective models do (e.g., Conati & Maclaren, 2009) and so we will extend and/or redesign them as needed.

In order to calibrate the main ICS creativity model, as well as its knowledge, affect, and meta-cognition sub-models, we will conduct empirical studies to collect data from students (high school and college) as they interact with the target tutor while a target set of sensors captures their physiological responses. The goal behind these evaluations will be to collect a rich data set that enable us to (1) evaluate the accuracy of the student models for capturing the target student attributes and (2) analyze how student actions and student affect influence the creative process during open-ended problem solving.

Model Accuracy: To determine student model accuracy we will compare model output to a gold standard (Arroyo et al., 2009; D'Mello & Graesser, 2012; Muldner, Burleson & VanLehn, 2010). In the case of student knowledge, this gold standard is typically a test targeting the domain concepts. For affect and metacognition, the situation is more complicated since information on students' feelings and high-level thoughts is not readily available. Thus, we will use a two-prong approach that we have relied on in the past: (1) talk-aloud protocol by having students verbalize their thoughts and feelings (e.g., Muldner et al., 2010) for a subset of the participants (since this is a laborious process that requires transcription and analysis of many fine-grained events), and (2) for obtaining affect information, the target system will intermittently ask students to report on their emotions (as in Arroyo et al., 2009). Note that these techniques are only necessary during model-testing – once the model is calibrated, the self-report prompts and talk-aloud protocol are removed. To use these data to assess model accuracy, we will transcribe the talk-aloud protocols and identify metacognitive and affective events, and then use these data in conjunction with the self-report data to compare against the corresponding submodel output.

Factors Influencing Creativity: While work in psychology has provided indications of how various attributes influence creativity, the technological context of this approach affords opportunities for investigating creativity beyond traditional settings. In particular, the PI's suite of sensors provides a unique chance for extending the community's knowledge on factors that influence creativity. Thus, we will rely on the EDM techniques we have used in the past (Muldner, Burleson, Van de Sande & VanLehn, 2011) and/or adopt additional ones as needed in order to mine the rich data set collected in this phase for factors influencing creativity. Specifically, relevant features will be extracted, e.g., affective states, productivity during problem solving, effort invested, and used as inputs to EDM techniques, e.g., Bayesian network parameter learning (Muldner et al., 2011) andlogistic regression (Cooper et al., 2009; Cooper et al., 2010). This will inform how various events contribute to creativity (e.g., a student reported frustration and this was related to a low creativity time span) and the relative utility of each event to the overall creativity process. We also plan to analyze the relative utility of each sensor (as we did for Muldner et al. [2010] and Cooper et al. [2010]) in order to understand which sensors are most valuable for creativity assessment as well as what the trade offs are when not all sensors are available.

Realizing Creativity Support within the GIFT architecture

As we have described above, ICS requires modeling of a range of student attributes, from domain knowledge, to meta-cognition, to affect. Aspects of the GIFT architecture are well aligned to support these modeling requirements. In particular, this architecture includes the *sensor module* that provides an interface for incorporating a range of sensing devices, which prior work has been shown to be useful for modeling affect (e.g., Arroyo et al., 2009). The input from these devices can then be sent to the GIFT *learner module* in order to map the low level sensor signals to the high level affective states of interest, like interest, frustration and/or flow – this module can also be used to assess students' domain knowledge and meta-cognitive skills. The GIFT *pedagogical modules* can rely on this information to tailor interventions in order to support and foster students' creativity.

GIFT also includes a *domain module*, that is used to structure and represent the target domain knowledge the student is expected to acquire – this is also relevant to creativity support, as students are expected to learn about the domain through creative activities. However, one aspect that is not clear and will need future exploration is how well the GIFT domain modules support the more open-ended domains that are required for creative endeavors, i.e., domains that afford users opportunities freedom to explore multiple solutions, apply divergent thinking and exhibit flexibility in their approaches. Many open-ended domains are ill defined in that it is difficult to specify objective criteria for solution evaluation – consequently we foresee this as one of the challenges in realizing creativity support in general and within the GIFT architecture in particular.

Conclusion

We have discussed the ICS framework and its application to the integration of ITS, affective computing, and CST to foster students' creativity. We have also outlined our research strategies for taking the next steps to implement and evaluate this approach, as well as initial considerations on how ICS can be realized within the GIFT architecture and challenges associated with doing so.

References

Aleven, V., McLaren, B., Roll, I. & Koedinger, K. (2006). Toward Meta-Cognitive Tutoring: A Model of Help-Seeking with a Cognitive Tutor. *International Journal of Artificial Intelligence in Education (IJAIED), 16*(2), 101- 128.

Amabile, T. M. (1983). The social psychology of creativity: A componential conceptualization. *Journal of Personality and Social Psychology, 45*(2), 357-376.

Amabile, T. M., Barsade, S. G., Mueller, J. S. & Staw, B. M. (2005). Affect and Creativity at Work. *Administrative Science Quarterly, 50*(3), 367-403.

Anderson, J. R. (1993). *Rules of the Mind*: Psychology Press.

Arroyo, I., Cooper, D., Burleson, W., Muldner, K. & Christopherson, R. (2009, July, 2009). *Emotion sensors go to school.* In Proceedings of the 14th International Conference on Artificial Intelligence in Education (AIED'09), Brighton, England.

Arroyo, I., Cooper, D. G., Burleson, W., Woolf, B. P., Muldner, K. & Christopherson, R. (2009). *Emotion Sensors Go To School.* In Proceedings of the 14th International Conference on Artificial Intelligence in Education (AIED'09), Brighton, UK.

Bielaczyc, K., Pirolli, P. L. & Brown, A. L. (1995). Training in Self-Explanation and Self-Regulation Strategies: Investigating the Effects of Knowledge Acquisition Activities on Problem Solving. *Cognition and Instruction, 13*(2), 221-252.

Cheung, C. K., Roskams, T. & Fisher, D. (2006). Enhancement of Creativity through a One-Semester Course in University. *The Journal of Creative Behavior, 40*(1), 1-25.

Clapham, M. M. (1997). Ideational Skills Training: A Key Element in Creativity Training Programs. *Creativity Research Journal, 10*(1), 33-44.

Cohen, P. A., Kulik, J. A. & Kulik, C. L. C. (1982). Educational Outcomes of Tutoring: A Meta-analysis of Findings. *American Educational Research Journal, 19*(2), 237-248.

Conati, C., Gertner, A. & VanLehn, K. (2002). Using Bayesian networks to manage uncertainly in student modeling. *User Modeling & User-Adapted Interaction, 12*(4), 371-417.

Conati, C. & Maclaren, H. (2009). Empirically Building and Evaluating a Probabilistic Model of User Affect. *User Modeling and User-Adapted Interaction*.

Cooper, D., Arroyo, I., Woolf, B., Muldner, K., Burleson, W. & Christopherson, R. (2009). *Sensors Model Student Self Concept in the Classroom.* In Proceedings of the UMAP 2009, First and Seventeenth International Conference on User Modeling, Adaptation and Personalization, Trento, Italy.

Cooper, D. G., Muldner, K., Arroyo, I., Woolf, B. P., Burleson, W. & Dolan, R. (2010). *Ranking Feature Sets for Emotion Models used in Classroom Based Intelligent Tutoring Systems.* In Proceedings of the International Conference on User Modeling and Adaptive Presentation (UMAP'10), Hawaii.

Corbett, A. T., McLaughlin, M. S. & Scarpinatto, K. C. (2000). Modeling student knowledge: Cognitive tutors in high school and college. *User modeling and user-adapted interaction, 10*, 81-108.

D'Mello, S. & Graesser, A. (2012). Language and Discourse are Powerful Signals of Student Emotions during Tutoring. *Learning Technologies, IEEE Transactions on.*

Dewett, T. & Gruys, M. L. (2007). Advancing the case for creativity through graduate business education. *Thinking Skills and Creativity*, *2*(2), 85-95.

Florida, R. (2002). *The rise of the creative class*. New York, NY: Basic books.

Isen, A. (2004). Some Perspectives on Positive Feelings and Emotions: Positive Affect Facilitates Thinking and Problem Solving. In N. F. a. A. F. Antony Manstead (Ed.), *The Amsterdam Symposium*. New York: Cambridge.

Isen, A. M., Daubman, K. A. & Nowicki, G. P. (1987). Positive affect facilitates creative problem solving. *Journal of Personality and Social Psychology, 52*(6), 1122-1131.

Kaufman, J. C. & Beghetto, R. A. (2009). Beyond big and little: The four c model of creativity. *Review of General Psychology, 13*(1), 1-12.

Koedinger, K. R., Anderson, J. R., Hadley, W. H. & Mark, M. A. (1997, August 16-19). *Intelligent Tutoring Goes To School in the Big City.* In Proceedings of the 7th World Conference on Artificial Intelligence in Education, Washington, DC, USA.

Lepper, M. R. (1988). Motivational Considerations in the Study of Instruction. *Cognition and Instruction, 5*(4), 289-309.

Ma, H.H. (2006). A Synthetic Analysis of the Effectiveness of Single Components and Packages in Creativity Training Programs. *Creativity Research Journal, 18*(4), 435-446.

Mack, R. W. (1987). Are Methods of Enhancing Creativity Being Taught in Teacher Education Programs as Perceived by Teacher Educators and Student Teachers? *The Journal of Creative Behavior, 21*(1), 22-33.

Mayer, R. E. (1999). Fifty years of creativity research. In R. J. Sternberg (Ed.), *Handbook of Creativity*: Cambridge University Press.

McCorkle, D. E., Payan, J. M., Reardon, J. & Kling, N. D. (2007). Perceptions and Reality: Creativity in the Marketing Classroom. *Journal of Marketing Education, 29*(3), 254-261.

Mitrovic, A. (2012). Fifteen years of constraint-based tutors: what we have achieved and where we are going. *User Modeling and User-Adapted Interaction, 22*(1-2), 39-72.

Muldner, K., Burleson, W., Van de Sande, B. & VanLehn, K. (2011). An Analysis of Students' Gaming Behaviors in an Intelligent Tutoring System: Predictors and Impacts, User Modeling and User Adapted Interaction. *Journal of Personalization Research, Special Issue on Data Mining for Personalized Educational Systems*, 99-135.

Muldner, K., Burleson, W. & VanLehn, K. (2010). "Yes!": Using Tutor and Sensor Data to Predict Moments of Delight during Instructional Activities. In Proceedings of the User Modeling, Adaptation, and Personalization Conference (UMAP'10).

NGA & CCSSO. (2012). Common Core State Standards Initiative: Preparing America's Students for College & Career, from http://www.corestandards.org

Picard, R. (1997). *Affective Computing*. Cambridge, MA: MIT Press.

Reye, J. (2004). Student Modeling based on Belief Networks. International Journal of Artificial Intelligence in Education (IJAIED), 14, 1-33.

Robbins, T. L. & Kegley, K. (2010). Playing with Thinkertoys to build creative abilities through online instruction. *Thinking Skills and Creativity, 5*(1), 40-48.

Runco, M. A. (2004). Creativity. *Annual Review of Psychology, 55*, 657-687.

Scott, G., Leritz, L. E. & Mumford, M. D. (2004). The effectiveness of creativity training: A quantitative review. *Creativity Research Journal, 16*(4), 361-388.

Self, J. (1998). The defining characteristics of intelligent tutoring systems research: ITSs care, precisely. *International Journal of Artificial Intelligence in Education, 10*, 350-364.

Shute, V. & Psotka, J. (1996). Intelligent tutoring systems: Past, present, and future *Handbook of research for educational communications and technology*. New York: Macmillan.

Trilling, B. & Fadel, C. (2009). *21st Century Skills: Learning for Life in Our Times*. San Francisco: Jossey-Bass.

Tripathi, P. & Burleson, W. (2012). *Predicting Creativity in the Wile: Experience Sample and Sociometric Modeling of Teams.* In Proceedings of the Conference on Computer Supported Cooperative Work (CSCW'12), Seattle, WA.

Vance, E. (2007). Colleges should teach broader skills to prepare students for work force, report says. *The Chronicle of higher education*.

VanLehn, K. (1988). Student modeling. In M. Polson & J. Richardson (Eds.), *Foundations of Intelligent Tutoring Systems* (pp. 55-78). Hillsdale, NJ: Erlbaum.

VanLehn, K., Lynch, C., Schulze, K., Shapiro, J. A., Shelby, R., Taylor, L., Wintersgill, M. (2005). *The Andes Physics Tutoring System: Five Years of Evaluations.* In Proceedings of the Artificial Intelligence in Education Conference (AIED'05).

Vygotsky, L. S. (2004). Imagination and Creativity in Childhood. *Journal of Russian & East European Psychology, 42*(1), 7-97.

Wince-Smith, D. L. (2006). The creativity imperative: A national perspective (Vol. 8, pp. 12 - 14).

CHAPTER 20 –Learner Modeling Considerations for a Personalized Assistant for Learning (PAL)

Damon Regan[1], Elaine M. Raybourn[2], and Paula J. Durlach[3]
[1]The Tolliver Group, Inc., supporting the ADL Initiative, [2]Sandia National Laboratories, and
[3]Advanced Distributed Learning Initiative (ADL)

Introduction

The Advanced Distributed Learning Initiative (ADL) has a vision for a Personalized Assistant for Learning (PAL), which will help provide life-long, relevant, tailored, timely access to learning content and performance aids. It will recommend topics and engagement modes appropriate to the learner's current context, and guide and support the user's learning experiences. GIFT and PAL have compatible goals in that both favor modular, service-oriented architectures, and learner-centric approaches (Sottilare, Brawner, Goldberg, and Holden, 2012). Learner models used by GIFT and PAL should predict learner state so needs can be best addressed by each system. Where GIFT and PAL differ is with respect to emphasis on adaptive support during learning versus adaptive recommendation of learning activities. Present day intelligent tutors use data from student-system interactions to build a learner model, and use that model to support learning activities, but each tutor confines itself to guiding learning in a limited domain and is pretty much ignorant of the user's context or their knowledge of other domains. In contrast, the PAL will need to aggregate data from student interactions with multiple systems and sensors in order to make context-aware recommendations about which learning activities to engage in. This chapter explores the challenges with establishing interoperability across sources of student data, so that the PAL learner model can use this information to provide assistance tailored to user preferences, goals, knowledge, and interests. These challenges are relevant to GIFT, because although not currently implemented, GIFT also intends to have a persistent learner model that provides a cross-domain view of the learner's performance, experience, and preferences.

Today, our learning opportunities extend well beyond traditional classroom education or on-the-job training. Recent years have seen the growth of educational technology such as distributed online content, serious games, simulations, MOOCs, and ITSs. Moreover, the distinction between formal and informal learning has become blurred through the availability of online resources such as Wikipedia and YouTube. There are applications that support online discussions, in which people post questions, obtain advice, and have discussions with both human and automated responders. Information pushed to users can be personalized through rich site summary (RSS) subscription feeds and aggregators; and social bookmarking allows users to add, annotate, edit, and share web documents. Learners "follow" the bookmarks, blogs, and posts of others. Digital tools also help people keep notes on, and products of, their activities (e.g., e-portfolios). The advent of cloud computing and mobile devices has freed these activities from the desk. In fact, there are so many possibilities available that learners may feel overwhelmed (Kuflik, Kay, and Kummerfeld, 2012). One of the overwhelming aspects is that each application or resource used by the learner knows little to nothing about what the user really needs. Some applications have the ability to collect user-system interaction data in order to alter or adapt future interactions (personalization), but since these tend to consider only the data they capture themselves, the picture of the user they create is just a tiny snapshot. Moreover, these adaptive systems tend to know little about the user's current context. True personalization would take into account some knowledge of the user's current situation and goals, and how these have changed overtime – context awareness.

Context awareness may encompass factors like location, task, role, and time. Recent developments in device and environmental sensors (ubiquitous computing) offer the promise of providing digital applications with some of this information. Coupled with a broader appreciation of learner knowledge, preferences, and interests, gleaned from user interactions with multiple applications, there is the potential to provide learners with just-in time content highly matched to their immediate requirements.

With respect to learning, what is of most immediate use and relevance depends on what the learner needs to know, why the learner needs to know it, and what the learner already knows. ITS and other adaptive instructional technologies are the best examples of learning technologies that attempt to provide what the learner really needs. By assessing performance of the specific skills the learner is trying to master, such technologies build up a model of learning gains and learning needs in a particular domain. But ITSs are closed systems. They only know how to instruct on a fixed skill set, using a limited array of teaching tactics. If that doesn't work, the learner is out of luck. They don't know how to reach out to find other resources (e.g., when it turns out, for example, that the user needs a refresher on prerequisite knowledge), and they only have a limited way of teaching. Why can't an ITS go out and find the user material on the prerequisites, or find other ways of teaching the same material when its own teaching tactics don't seem to be working? What about when the user really doesn't want a lesson, but just needs to get an answer quickly? How can we go from the situation of today, where each application we use has only a little bit of the picture – like the proverbial blind men feeling the elephant – to the situation in which these pieces of the picture can be amalgamated to produce more context-aware and personalized support?

Future Concept

ADL is investigating solutions to these challenging questions, under the PAL project. ADL intends for a PAL to enable learners to access personalized learning content and peer and mentor networks from multiple devices or platforms. It will track performance throughout the learning process, whether formal or informal. It will have a nonintrusive user interface and will consume data from device and environmental sensors to support context awareness. In the short to mid term, we envision core PAL services providing a user interface, experience tracking, user modeling, recommendation formulation, and content discovery. In the longer term, we envision that a PAL could also provide coaching and guidance. Thus, whereas GIFT has started out like an ITS (coaching) and will expand to encompass a persistent student model that spans domains, the PAL is starting out like a recommender system. A PAL will be knowledgeable about diverse domains and provide the appropriate level of guidance for the user's current needs and goals. A PAL should help create a self-regulated learner with a thirst for knowledge, and not engender dependency on technology.

PAL Scenario

Consider the scenario of an U.S. Army Reserve soldier in 2020. Corporal Smith is about to deploy to Haiti on a hurricane relief mission. Since her PAL is linked to the Army's personnel system, it already knows she is preparing to deploy and has recommended to her several relevant training exercises. Her PAL knew that she already had a badge for Level 2 French (limited working proficiency) according to the language competency criteria set by the Interagency Language Roundtable (ILR), and that she wants to get to Level 3 (professional working proficiency). Her PAL recommended some lessons and game-based scenarios so she could brush up on her old skills and learn some Haitian Creole. Her PAL knows that her main job while deployed will be managing the X22 search and rescue robot, so it has selected new vocabulary that will likely be useful in that situation. It also cued up the interactive technical manual for the X22. On the flight to Haiti from Kansas, her PAL offers her various methods of passing the time. She selects a documentary about the impact of the 2017 election to Pope of Archbishop of Port Au Prince, and she reviews the latest footage and reports available about the hurricane damage. Once on the ground, with a little free time, Corporal Smith networks with some colleagues back home who also have experience with the X22. They all know that the X22 sensors have a tendency to malfunction in humid conditions and are discussing possible solutions. The corporal's PAL suddenly alerts her network to a new blog it has found discussing the exact same issue, with an offered solution, but will she really find BOUNCE® in Haiti now? She wishes she had thought to investigate this issue before she left and asks the PAL for the Haitian Creole translation for dryer sheets.

Omnipotent ITS or Interoperable Systems?

This vision of a PAL is of an intelligence that can make recommendations and monitor user activities in real time. It has the ability to infer the user's needs and respond by making recommendations for learning, and it also is ready to provide information in the form of assistance (e.g., to provide the translation for dryer sheets). Today's ITS are not capable of this kind of support, because they are limited to intelligence about learning a particular domain, and they also tend to have a limited way of interacting with the student (e.g., designed to teach how to solve linear equations, but not prepared to help answer a real-world problem the student needs to solve right now). It might be possible to grow an ITS into a PAL, by expanding its breadth of expertise as well as the various services it provides (e.g., supporting social networking, providing references, etc.). It might even be possible to create an ITS that can ingest information from the web and "learn" new domains. Given the challenges and resources required for creating an ITS for just one domain, this seems like a very daunting task. In addition, users will still always have other activities that the ITS will be ignorant of. Finally, it is unclear how a monolithic system could ever keep up with changing technologies. An alternative approach is to foster interoperability among existing and future heterogeneous systems, all of which can contribute some knowledge of the user to a persistent learner profile (Alrifai, Dolog & Nejdl, 2006; Carmagnola, Cena & Gena, 2011; Kay & Kummerfeld, 2012; Kulik, Kay & Kummerfeld, 2012). This learner profile would exist independent of any service or application, allowing those services and applications to both contribute and consume its knowledge about the learner. As discussed by Kay and Kummerfeld (2012), there are many technical challenges associated with persistent learner profiles. These include the learner's ownership of their own data, and their ability to inspect them, as well as privacy control and protection of personal data. Our focus in this chapter, however, is on the interoperability challenge. How can we unlock the data from heterogeneous systems and make open data, content, and web APIs the new default?

A Service-Based Approach

One potential solution to the problem of data being locked up inside of applications is to securely expose the data through web APIs using common standards so that multiple applications can use the data (CIO Council, 2012). While it is clear that learner data need to be exposed securely through web APIs, it is not yet clear how that learner data should be represented. Approaches to represent interoperable learner data have historically focused on XML and resource description format (RDF) standards in the W3C Semantic Web stack (IMS, 2001; Dolog & Schafer, 2005). While these approaches have relied on standards, the barrier to entry appears to be too high, perhaps due to the complexity of requiring all contributors to use a common schema. Alternative approaches that exploit hypertext transfer protocol (HTTP), JavaScript object notation (JSON), and Representational State Transfer (REST) may better facilitate the feasibility of communication and the exchange of learner data (Carmagnola, Cena & Gena, 2011). The reason HTTP, JSON, and REST are important is that they are the simple and lightweight methods that web developers prefer to get started with – they lower the barrier to entry.

The Mozilla Open Badges project (http://openbadges.org/) provides a new way to think about representing learner data that exploits HTTP, JSON, and REST. Mozilla Open Badges make it possible to get recognition for skills and achievements that happen online or out of school by making it easy for any organization to issue, manage, and display digital badges across the web. The Open Badges project includes a badge format specification, APIs, and a reference implementation for "badge backpack" software. The badge backpack software, which keeps track of badges a user has been awarded, is a critical collection of learner model data. Various organizations that may issue badges can choose the granularity of a given badge.

The ADL Experience API (http://www.adlnet.gov/tla/experience-api) is another project providing a new way to think about representing learner data that also exploits HTTP, JSON, and REST. The Experience API provides a way to track and report learning activity using simple statements. Together, Open Badges and the Experience API offer a new way to think about constructing interoperable learner model data.

To illustrate how these new projects may be used together, consider how badges may be issued. A learning application can provide direct credit for a learner's achievement using that platform in the form of a badge. However, this is not the only way a learner could be given credit. The Experience API allows many different learning experiences to be tracked and reported at a very granular activity level (e.g., Sarah watched Army Knowledge Online Video X). This activity data, for a given user, can be collected and securely made available through a Learning Record Store (LRS; http://www.adlnet.gov/tla/lrs). The data can then be analyzed independent of the learning experience. An assessment service for example, using specialized techniques, could examine a learner's activity data stored in an LRS and provide credit (e.g., a badge) for relevant activity that may span different applications. Figure 20-11 shows these two techniques for providing a learner with a badge.

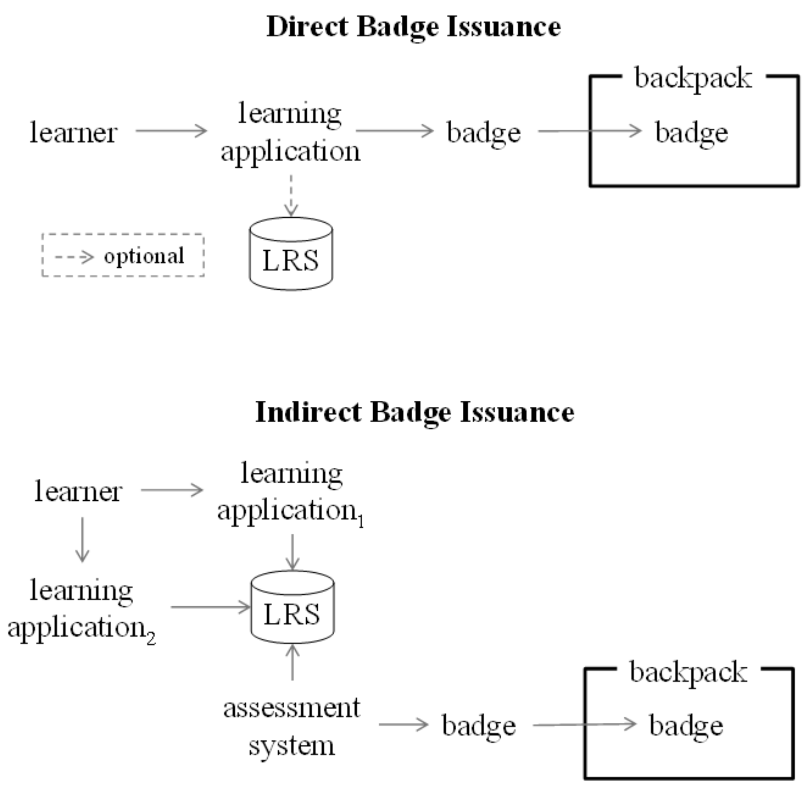


Figure 20-1. Direct and indirect badge issuance. For direct badge issuance, a learning application assesses learning activity and puts a badge in the learner's badge backpack when the activity reaches an objective criterion. That learning application may or may not also send information about the learner's activity to a separate LRS. For indirect badge issuance, an assessment system examines data in the LRS. Those data may have been contributed by one or more learning applications. The assessment system puts a badge in the learner's backpack when the evidence in the LRS reaches an objective criterion.

Open Badges and the Experience API also provide a way to capture *evidence* for achievements. In Open Badges assertions (Cmcavoy, 2013), there is an optional field to track evidence for a given badge issuance. The evidence field is simply a uniform resource locator (URL) that should point to information about how the learner earned the badge. Given the Experience API and the collection of data in a web-accessible LRS, the Open Badges evidence field can point to specific experience activity data located in an LRS as depicted in Figure 20-22.

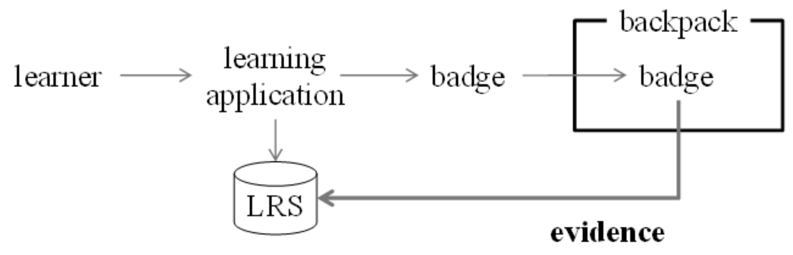


Figure 20-2. A specific badge could point to the specific activities in the LRS as the evidence concerning how the badge was earned.

The Experience API is the first web service specification and Open Source Software element in a collection of web service specifications and Open Source Software elements that will make up the ADL Training and Learning Architecture (TLA). In addition to services for tracking learning experiences, the TLA will also include services for managing learner profiles (http://www.adlnet.gov/tla). It is anticipated that Open Badges will be referenced by learner profiles, which will likely contain other learner data such as goals, reflection, etc. The TLA will also include services for creating and accessing competency definitions to serve as a common way to reference educational standards, learning objectives, and competency definitions through web APIs. The Open Badges assertion specification also contains a field to reference criteria for earning a badge. The criteria field is simply a URL that should point to the badge's criteria. Figure 20-3 illustrates how badge criteria could point to competency definitions. For example, a badge that the Khan Academy might issue for a learner mastering algebra could reference criteria for algebra using the Math Common Core standards accessible through the Achievement Standards Network's (ASN) web APIs.

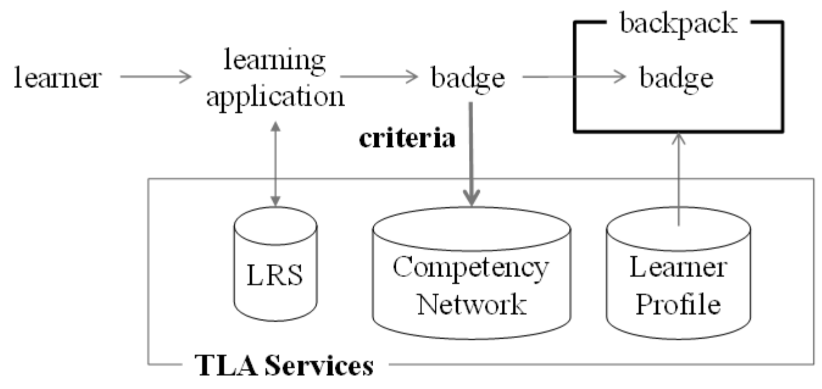


Figure 20-3. TLA services linking the learner profile to the badge backpack, and the badge itself to sanctioned competency criteria

A learner model used within an adaptive system can choose to model that learner data using any representations the developer of the application wants to use. However, interoperable and shared learner model data has to be represented and made available in standard ways to support as many applications being able to use the information as possible. If the standard representation is complex, this may enable some sophisticated applications to use very powerful representations of learner information. However, other less sophisticated applications may be unable to use the information because it is simply too complex. If the standard representation is simple, this will likely enable all applications to make maximum use of learner model data. However, in order to extract the power of the learner model data, specialized techniques will be required. The goal is find simple ways to share learner information that can be used and extended in powerful ways with significant community adoption.

Machine Interpretation

The effort to enable machines to access as much shared heterogeneous learner model data as possible will likely require making the effort to contribute that data from many applications as easy as possible. This means removing as many complex constraints on contributors as possible. A significant constraint past approaches have imposed on applications that contribute learner profile data is the requirement to conform that data to common sophisticated representations based on Semantic Web technologies. Simplified alternative approaches that impose less complex constraints are likely to make contributions easier and promote greater adoption.

While the approach of removing as many complex constraints on contributors as possible may be good for contributors, it may make it difficult for consumers of data. Flexibility on publishing leads to complexity on consumption, where the machines have to be able to make sense of the data (i.e., translation). However, this seems to be as it should be, for each application that consumes learner model data will likely want to use data in a particular manner for a particular application. While the skills required to make sense of big data may be currently scarce and data science is a nascent field, it is taking off in a big way, with educational offerings growing and the pipeline of talent expanding (Davenport & Patil 2012).

The method for making sense of heterogeneous learner model data has been described as bootstrapping learner models from external sources (Guo & Greer, 2007; Tiroshi, Kuflik, Kay & Kummerfeld, 2011). Guo and Greer (2007) bootstrapped (i.e., created and initialized) learner models from e-portfolios for an advanced programming course. The method of bootstrapping Guo and Greer (2007) employed involved experts and students linking of e-portfolio evidence for skill level claims in the learner model using a custom user interface and comparing their estimations. The system provided an evidence-recommending feature when a given e-portfolio artifact was selected. It should be noted that during bootstrapping, the raw experience data stored in a LRS referenced by an Open Badge, could be analyzed to perform an application's own interpretations of that evidence, as suggested by Carmagnola, Cena, and Gena (2011). Guo and Greer (2007) concluded that the bootstrapping method showed promising benefits in adaptive learning environments. Tiroshi, Kuflik, Kay, and Kummerfeld (2011) summarized various boostrapping methods (i.e., content analysis, machine learning, and sentiment analysis) to initialize user models for recommender services from social web APIs and proposed a theoretical example of using the users' social data to provide personalized tours tailored to users' interests.

To better understand the potential for the vision of bootstrapping learner model data, consider its application to a different kind of data. The Learning Registry is a distributed database for different educational sites to share metadata about educational resources and information about how resources are being used in different educational settings. The approach to make contributing as easy as possible was similar to the approach being described here of making it as easy as possible for applications to publish learner model data. Thus, there were minimal constraints on the form metadata for resources needed to take – contribution focused on flexibility.

The flexibility afforded contributors made it very difficult for consuming applications to use the large volume of heterogeneous data, however. To address this difficulty, ADL developed an approach to make sense of the data through an open source project called lr-data (https://github.com/wegrata/lr-data), which has subsequently been shared with and used by others. The technique is based on a distributed task queue called celery (https://github.com/celery/celeryproject). Different tasks are assigned to process each record of data (harvest it, clean it, filter it, augment it, store it, etc.). The beauty of the software is that it is open source, visible (i.e., hosted on github), and written in the easy to read language of python. Different applications can use and adapt this open source toolkit to process the data as they need for their specific application requirements.

In the case of sharing and making sense of learner data, a similar model can be followed where any contributing application (e.g., an ITS or game) could easily publish activity data to a LRS, Open Badges to a learner's badge backpack, or other learner model data to their shared learner profile as JSON over a RESTful HTTP connection. The LRS, badge backpack, or learner profile can exist anywhere (e.g., an employer organization, a commercial provider, or even self-hosted by a learner) – all the contributing applications need to know are the URLs and the learners credentials. If a learning application wants to make sense of all this data, it too just needs the URLs and the learner's credentials to get started. The learner could set permissions concerning what applications can access their data. Then a bootstrapping method can be used to make sense of the data leveraging open source software customized for the needs of the learning application. The software would likely reference external ontologies, databases, and services. Finally, a learner could ideally be involved in running such software to make sense of their own data for their own purposes.

Discussion

The PAL, GIFT, and other learner applications that will exist in the future will benefit from sharing learner model data with one another to improve the way each learning application can best adapt to the learner. A flexible approach based on web services was proposed.

The data sharing design goals of PAL and GIFT are complementary. Each seeks to monitor progress and provide tailored assistance by accessing heterogeneous data sources. For example both GIFT and PAL offer integration of domain-specific data and domain-independent modules or plug-ins to contribute to learner models. Key to both systems is the notion of data sharing via interoperable open standards. Data and data structures in GIFT reside within models and libraries, while executable code is expressed as interoperable modules (Sottilare, Brawner, Goldberg, and Holden, 2012). We are considering how PAL can use elements of GIFT, but also how PAL might bootstrap learner data that resides anywhere (as long as access permissions exist).

To support comprehensive and persistent learner models to enable the adaptive tutoring learning effect chain (Sottilare, 2012), GIFT should plan to contribute and consume from external shared learner model infrastructure such as badge backpacks, LRSs, and learner profiles. GIFT should plan to participate in open source development of toolkits to support bootstrapping external learner data as part of the design of the GIFT authoring, instructional management, and assessment constructs for more flexible data-driven computer-based training systems (CBTS).

As we struggle to understand which data are most important in a learner model for the desired learning outcomes (Holden, Sottilare, Goldberg & Brawner, 2012), having access to as much data as possible is critical to conduct the anticipated research. Given that GIFT is particularly focused on the needs of adult learners (Holden, Sottilare, Goldberg & Brawner, 2012), capturing an understanding of their diverse experiences seems paramount.

Finally, while much of the focus on GIFT learner modeling is on the current tutoring session and the learner's state in that session, it is important to keep in mind how the learner evolves between tutoring sessions. When the learner begins a subsequent tutoring session, it would be valuable for a GIFT tutor to know what the learner may have experienced and learned since the last tutoring session.

Acknowledgements

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy's National Nuclear Security Administration under Contract DEAC04-94AL85000.

References

Cmcavoy (2013, March 13). Assertions – mozilla/openbadges Wiki. Retrieved March 15, 2013 from https://github.com/mozilla/openbadges/wiki/Assertions

Alrifai, M., Dolog, P., and Nejdl, W. (2006). Learner profile management for collaborative adaptive elearning applications. Proceedings from APS '06: The International Workshop on Adaptivity, Personalization, and the Semantic Web. New York, NY: ACM. Retrieved March 15, 2013 from: http://people.cs.aau.dk/~dolog/pub/aps2006.pdf

Carmagnola, F., Cena, F., and Gena, C. (2011). User model interoperability: a survey. User Modeling and User-Adapted Interaction, 21(3), 285-331.

CIO Council (2012). Digital government: Building a 21st century platform to better serve the American people. Retrieved March 15, 2015 from: http://www.whitehouse.gov/sites/default/files/omb/egov/digital-government/digital-government.html

Davenport, T. H. and Patil, D. J. (2012). Data scientist: The sexiest job of the 21st century. Harvard Business Review. Retrieved March 15, 2013 from http://hbr.org/2012/10/data-scientist-the-sexiest-job-of-the-21st-century/ar/1

Dolog, P. and Schäfer, M. (2005). Learner modeling on the semantic web. Proceedings from PersWeb '05: Personalisation on the Semantic Web at User Modelling 2005: 10th International Conference. Retrieved March 15, 2013 from http://www.win.tue.nl/persweb/Camera-ready/6-Dolog-full.pdf

Guo, Z. and Greer, J. (2007). Bootstrapping accurate learner models from electronic portfolios. Proceedings from Artificial Intelligent in Education '07: Building Technology Rich Learning Contexts That Work. Amersterdam, The Netherlands: IOS Press.

Holden, H.K., Sottilare, R.A., Goldberg, B.S., Brawner, K.W. (2012). Effective learner modeling for computer-based tutoring of cognitive and affective tasks. I/ITSEC 2012 Proceedings, Interservice/ Industry Training, Simulation and Education Conference, Orlando, Florida, USA.

IMS (2001). IMS learner information package specification. Retrieved March 15, 2013 from: http://www.imsglobal.org/profiles/lipinfo01.html

Kay, J. and Kummerfeld, B. (2012). Lifelong learner modeling. In P. Durlach & A. Lesgold (Eds.), Adaptive Technologies for Training and Education (pp. 140-164). New York: Cambridge University Press.

Kuflik, J. Kay, and B. Kummerfeld (2012). Challenges and solutions of ubiquitous user modeling. In A. Kruger & T. Ksvi (Eds.), Ubiquitous Display Environments (pp. 7-30). New York: Springer. Retrieved March 15, 2013 from: http://sydney.edu.au/engineering/it/~judy/Homec/Pubs/2012_Ubiquitous_User_Modeling.pdf

Sottilare, R. (2012). Considerations in the development of an ontology for a Generalized Intelligent Framework for Tutoring. *International Defense & Homeland Security Simulation Workshop in Proceedings of the I3M Conference*. Vienna, Austria, September 2012.

Sottilare, R.A., Brawner, K.W., Goldberg, B.S. & Holden, H.K. (2012). *The Generalized Intelligent Framework for Tutoring (GIFT)*. Orlando, FL: U.S. Army Research Laboratory – Human Research & Engineering Directorate (ARL-HRED).

Tiroshi, A., Kuflik, T., Kay, J., Kummerfeld, B. (2011). Recommender systems and the social web. Proceedings from UMAP '11: The 19th international conference on Advanced in User Modeling (pp. 60-70). Heidelberg, Berlin: Springer.

CHAPTER 21 –Eye-Tracking for Student Modelling in Intelligent Tutoring Systems

Cristina Conati[1], Vincent Aleven[2], and Antonija Mitrovic[3]
[1]University of British Columbia, Vancouver, Canada, [2]Carnegie Mellon University, Pittsburgh, U.S.A, and [3]University of Canterbury, Christchurch, NZ

Introduction

In this chapter, we review research on leveraging eye-tracking information to improve the depth and accuracy of student modeling in ITSs. Eye-tracking has been extensively used both in psychology for understanding various aspects of human cognition, as well as in HCI for offline evaluation of interface design or as an alternative form of intended user input. In recent years, however, eye-tracking has also been investigated as a source of information on relevant users states and processes (e.g., attention, motivation, meta-cognitive activity) to inform the actions of an ITS. This chapter is an overview of some of the recent trends in this area. The overview is structured in two parts. In the first part, we describe existing research on investigating eye-tracking data as a direct source of information for student modeling and personalized instruction. In this part, we discuss efforts to model the learner at the behavioral, cognitive, meta-cognitive and affective level. The second part focuses on research that has used eye-tracking mainly for the offline analysis of how students attend to specific elements of an ITS interface, in order to understand relevant student behaviors and processes. Although this work is less directly related to using eye-tracking data in student modeling than the work described in the first part, the results of this research provide important insights on additional ways in which student models could leverage gaze-data in the future. We conclude the chapter with a discussion of these insights and related recommendations for GIFT design.

Investigating Gaze Data as a Direct Source Of Information for Student Modeling

Leveraging Eye-Tracking Data to Capture and Adapt to Relevant Student Attention Behaviors

The work by Sibert et al. (2000) represents, to our knowledge, the first attempt to use gaze tracking for real-time student assessment. Sibert et al. (2000) describe GWGazer Reading Assistant, a system for automated reading remediation that tracks a student's reading patterns and provides support if these patterns indicate difficulties in reading a word. In particular, raw gaze data tracked with an unobtrusive, camera-based eye-tracker is parsed in real time to identify the word a student is currently reading, based on a *reading dwell* threshold that essentially defines the minimum amount of attention needed to be focusing on a word. A second threshold identifies delays in dwelling on a word that may indicate difficulty in reading it. When dwelling on a word exceeds this second threshold, the Reading Assistant pronounces the word for the student as an aid to reading it. Sibert et al. (2000) describe a preliminary informal evaluation of the system, in which eight children age 10–14 read a series of textual passages twice, with the help of the Reading Assistant. Results are presented in terms of changes in reading speed, accuracy, and number of prompts received from the first to the second reading of each passage, showing improvements on all measures. While these results do not provide any formal conclusions on the effectiveness of the Reading Assistant gaze-drive audio prompts, a qualitative post-questionnaire revealed that students liked the system and found it easy to use and unobtrusive. Thus, this work can be seen as an encouraging preliminary step in developing learning environments that leverage gaze information to provide personalized support to their users.

Anderson (2002) conducted an early experiment with a gaze-contingent ITS. The study involved a research version of the Pump Algebra Tutor (PAT), created specifically for use in eye-tracking studies (e.g., the interface elements were spaced out more widely than in the standard tutor). The purpose of the experiment was theoretical, namely, to provide an existence proof that paying attention to "fine-grained temporal detail of the student's behavior" can have instructional leverage. More specifically, building on the work by Gluck and Anderson (2000), Anderson made the tutor respond to certain instructional opportunities that could be identified only through eye-tracking. As one example, if following an error message, the student did not read the error message (as revealed through eye-tracking) and did not correct the error within 10 seconds, the tutor would give a brief auditory message ("Read the help message"). As a second example, when – through eye-tracking – the tutor detected that the student used a problem-solving strategy that bypassed the instructional objectives, instead of using an intended problem-solving strategy, the tutor would give an auditory message suggesting that the student try the intended strategy (e.g., one in line with the instructional objectives). This happened in algebra problems in which students were given a problem statement describing a story context and were asked to (1) formulate an algebraic expression that captures the algebraic relations described in a story context and (2) use that expression to calculate specific quantities. On the second step (i.e., calculating quantities by applying the algebraic expression), eye-movement data revealed that students sometimes ignored the algebraic expression that they had created moments earlier and instead reasoned directly from the problem statement, a slower strategy that bypassed the objective of learning to work with algebraic expressions. When the student made an error on this type of step without fixating on the expression, the gaze-contingent tutor presented another auditory message: "Try using the formula to compute your answer." With these gaze-contingent additions, the tutor helped students reach mastery 20% faster than the standard tutor, an impressive gain in efficiency. Further, eye-movement data revealed that students who worked with the gaze-contingent tutor attended to the algebraic expression more even before the gaze-contingent tutor would suggest they do, evidence that the gaze-contingent messages had the desired result.

Wang et al. (2006) used eye data to control interaction with the Empathic Software Agents (ESAs) for teaching biology. Eye-tracking data were used in two ways: as user input and also to provide information for adapting the behavior of the pedagogical agents available in the ESA to the student. Using the gaze input, it was possible for the student to choose a topic to study by simply looking at the appropriate area on the screen for a pre-defined time period. The student could also reply to a yes/no question by using the appropriate eye gesture (moving the eyes vertically for "yes" and horizontally for "no"). By analyzing the eye-tracking data and pupil dilation, the system inferred the student's focus of attention and responded to it with affective behavior and/or feedback. When the student showed interest in a particular content by dwelling on it, the agent moved to the appropriate location on the screen and provided additional information. In addition, the agent also provided positive affective feedback to the student's attentiveness by showing facial expressions conveying happiness. In contrast, if the student appeared to lose concentration (e.g., by looking away from the screen), the agent would say something to bring the student's attention back to the screen and showed mild anger. The agents also displayed adaptive behaviors based on student's states inferred primarily from student actions, e.g., providing feedback if a student made a mistake, or trying to engage the student if she appeared bored or disengaged because of lack of mouse or keyboard input. A small preliminary study with 10 participants revealed a beneficial effect of the adaptive agents on the students' motivation and concentration. The participants reported that they were more attentive to additional information and explanations provided by the agent than to the other content available in the system. Although the study does not provide sufficient information to discriminate which role the gaze-adaptive components played in these evaluations, it shows that, overall, an agent relying on both gaze and action data to provide cognitive and affective feedback has good potential to enrich a student's learning experience.

D'Mello et al. (2012) provide the only demonstration so far (to the best of our knowledge) that an ITS responding to student gaze in real time can improve student learning. (The study by Anderson [2002], described above, demonstrated that adding gaze-contingent responses to an ITS can lead to more efficient learning, but not that it leads to better learning outcomes.) The study involved Guru, a dialogue system with an on-screen tutor agent that engages the student in a tutorial dialogue on instructional material displayed on the screen. The system used eye-tracking to evaluate whether the student was paying attention, as captured using simple rules (basically, not looking at the tutor agent or the relevant instructional material was considered not paying attention). When the student did not pay attention, the tutor interjected (in speech) any of the following messages: ''Please pay attention,'' ''I'm over here you know,'' ''You might want to focus on me for a change,'' and ''Snap out of it. Let's keep going.'' D'Mello and colleagues conducted a study comparing tutor versions with and without these gaze-responsive messages. The results were very interesting. First, there was substantial reorientation of gaze after the gaze messages, meaning that the tutor agent did succeed in directing students' attention back to the tutor agent and the lecture. Further, students who worked with the gaze-reactive tutor did better on deep learning questions on the post test than students who worked with the version that was not gaze-reactive. In contrast, learning gains for assertion questions in the pre-post test, which tap into knowledge of surface level facts, were higher with the non-gaze-reactive tutor.

All the work described in this section leverage gaze information to capture momentous student attention patterns relevant to improving student interaction with the corresponding learning environments (i.e., patterns indicating reading difficulty in the GWGazer Reading Assistant, and patterns indicating attention or lack thereof in ESA and Guru). None of this work, however, uses the captured gaze information to make higher-level inferences regarding student's states and processes. In the next section, we review work that takes this extra step, using gaze data to model students at the cognitive, meta-cognitive, and affective level.

Leveraging Eye-Tracking Data to Model Student Cognitive, Meta-Cognitive, and Affective States

Conati and Merten (2007) use gaze data to improve the accuracy of a student model designed to enable provision of personalized support to learning mathematical functions via exploration of an interactive simulation (Adaptive Coach for Exploration [ACE]). Providing this adaptive support is challenging because it requires assessing the effectiveness of behaviors for which there is no formal definition of correctness. Conati and Merten (2007) tackled the challenge with a probabilistic model that assesses exploration effectiveness by integrating information on (1) user actions in ACE, (2) user's knowledge and (3) whether users actually reason about (self-explain) their exploratory actions. Self-explanation – generating explanations to oneself to clarify instructional material –is a well-known meta-cognitive skill in cognitive science. This work is the first to consider and model self-explanation in the context of exploration-based learning. To assess whether a student is self-explaining the outcome of an exploratory action, the ACE's student model combines information on the time the student spent on that action with gaze information. This gaze information relates to the occurrence of a simple gaze pattern defined a priori as being relevant for learning with this particular simulation: a gaze shift between two panels, one showing a function equation and one showing the related plot. The main exploratory action available in this simulation is to change either the equation or the plot, and see how the change affects the other component. Hence the definition of the aforementioned gaze shift as a relevant pattern to indicate self-explanation in ACE. A formal evaluation showed that the student model including eye-tracking information provides significantly better assessment of both a student's self-explanation behavior during interaction with the simulation, as well as subsequent learning of the relevant mathematical concepts.

In the student model described above, Conati and Merten used gaze information related to the occurrence of a simple gaze pattern defined a priori as being relevant for learning with their target simulation. Kardan and Conati (2012) and Kardan and Conati (to appear) extend this work by looking at a much broader range of general eye-tracking features to capture student learning in the context of a different interactive simulation (IS). This is an important difference, because pre-defining gaze patterns that indicate learning in an IS may not always be easy or possible, due to the often unstructured and open-ended nature of the interaction that IS support. Furthermore, such pre-defined patterns are task specific, and may not directly transfer to a different IS. In contrast, the approach described in Kardan and Conati (2012) and Kardan and Conati (to appear) is more general and can be applied to a variety of ISs. It relies on giving to a classifier user model a broad range of standard eye-gaze features that are either task independent or based solely on identifying the main components of the target IS interface. Then, it is left to the classifier to identify patterns that are indicative of users' learning with that IS. An evaluation of this approach was performed on a data set encoding the gaze data of students working with the constraint satisfaction problem (CSP) applet, an IS designed to visualize the workings of the AC3 algorithm for constraint satisfaction on a variety of available sample problems. The CSP applet provides various functionalities that allow a student to explore the run-time behavior of AC3 at their own pace. The evaluation described by Kardan and Conati (2012) showed that a classifier using solely information on a student's overall attention patterns during a complete session with the CSP applet achieves an accuracy of 71% in distinguishing students who learned well from the CSP applet from students who did not (where learning was measured via a pre-test and post-test administered during the study). Furthermore, giving the classifier additional information on how students' attention patterns changed while solving two different problems of increasing difficulty further improved classification accuracy to 76%, with better balance in classifying each learner type (i.e., high learners vs. low learners, with class accuracy of 77% and 78%, respectively). In a follow-up study, Kardan and Conati (2013) showed that a student model for the CSP applet that combines information on both gaze data and interface actions outperforms models that rely on either gaze data or action data only. Kardan and Conati (to appear) also show that the action+gaze student model for the CSP applet reaches and stays above 85% accuracy in classifying a new user as a high versus low learner after seeing 22% of the overall interaction data (accuracy above 80% in each class), showing that the model can be used to trigger real-time interventions aimed at improving the experience of low learners with the CSP applet. Thus, Kardan and Conati's work provides further evidence of the value of gaze data for user modeling, especially for interactions in which it is hard to predefine a priory the learners' behaviors that should be detected as relevant or detrimental for learning.

Similar results were obtained by Bondareva et al. (2013), when using gaze data only to predict learning with a different type of educational environment, namely, a multi-agent ITS (known as Meta-Tutor), that scaffolds self-regulated learning (SRL) while students study science material (Azevedo et al., 2012). MetaTutor is an adaptive hypermedia learning environment, which includes 38 pages of text and diagrams, organized and accessible by an interactive table of contents. Text and diagrams are displayed separately in the two central panels of the interface. In addition to providing structured access to relevant content, MetaTutor also includes a variety of components designed to scaffold learners' use of SRL processes and their learning of a target science topic, e.g., the human circulatory system. Four pedagogical agents (PAs) provide spoken prompts and feedback on various SRL processes. For example, one PA assists the student in establishing two learning sub-goals related to the overall learning goal for the session. Other SLR processes supported by the PAs include taking notes, writing summaries of the viewed content, and evaluating one's current understanding via interactive quizzes.

The results in (Bondareva et al., 2013) show that, by leveraging gaze features similar to those used in Kardan and Conati (2013), a logistic regression classifier achieves 78% accuracy on predicting student learning with Meta-Tutor, after seeing all data from an interaction. Accuracy already reaches 72% accuracy after seeing 37% of the data. These results are especially important because, in combination with the results in Kardan and Conati (2013), they confirm the importance of gaze data as a predictor of learning *across different types of learning environments* that can be leveraged for providing real-time personalized support to student learning.

Qu et al. (2005) leveraged gaze data to assess student motivation in the Virtual Factory, an ITS that teaches engineering skills (Johnson, Rickel, and Lester, 2000). They started from observations that human tutors use information about a learner's motivational states related to effort, confusion, and confidence during coaching. Based on these observations, Qu et al. (2005) enhanced an animated pedagogical agent with the ability to infer the same motivational factors about students. Information about the student's interface actions as well as gaze data tracking a student's focus and duration of attention were used as input for a dynamic Bayesian model, which inferred a learner's confidence, effort, and confusion during interaction the Virtual Factory. This student model was tested through a Wizard of Oz study during which students were interacting with a version of the Virtual Factory with the PA's interventions being directed by an experimenter. During the study, log data were collected, along with videos of the students' face and student retrospective self-reports on their motivational states during interaction. Two judges labeled replays of each session, synchronized with the videos of the students' face, for confusion, effort, and confidence (as Low, Medium, and High). The student model's predictions over the three factors were compared against both the judges' generated labels and the students' self-reports, showing very encouraging accuracies between 70% and 82%. Thus, this work provides initial evidence that gaze information can help assess student's affective states in addition to more strictly cognitive factors.

Off-Line Analysis of Gaze Data to Understand Relevant Student Behaviors and Processes.

Seminal work by Gluck, Anderson, and Douglass (2000) demonstrated that eye-movement data of students working with an intelligent tutoring system contain information about students' cognitive processes that is not directly available from the regular stream of student-tutor interaction data (see also Anderson, 2002). By performing offline analysis of eye-tracking data obtained with a simplified version of the PAT algebra tutor (later named the Algebra Cognitive Tutor), these researchers were able to predict certain errors even before they happened. They also showed that eye-tracking data could quite reliably disambiguate domain-specific strategies even when they led to the same problem-solving steps. Finally, using eye-tracking, it became apparent that students did not attend to as many as 40% of the system's error feedback messages. Although the work by Gluck et al. (2000) did not actually demonstrate a method for updating a learner model based on eye-tracking, it is important for this survey because it clearly indicates the potential of gaze data as a rich source of information for student modeling, especially the strategy disambiguation work, in which inferences from eye-movement data to cognitive processes were made quite successfully, which often tends to be rather difficult step, fraught with uncertainty.

In relation to using gaze data to evaluate whether students attend to an ITS's adaptive interventions, Muir and Conati (2012) performed offline analysis of gaze data to investigate not only if, but also why students pay attention to adaptive hints generated by an educational game for math (Prime Climb). Prime Climb provides game activities to help students practice skills related to number factorizations, and includes a pedagogical agent that helps students learn from these activities by providing individualized hints. These hints are based on a student model that assesses whether students are learning during a session with Prime Climb, given their game actions. The hints are provided at incremental level of detail when the model predicts that student's knowledge of relevant factorization skills is low. The hints include (1) reminders to use available tools that can show how a number is factorized; (2) definitions of relevant factorization concepts, accompanied by illustrative examples; and (3) "bottom-out" hints that explicitly explain why a student action was correct or incorrect based on factorization knowledge.

Providing adaptive hints to support learning during game play is challenging because it requires a trade-off between fostering learning and maintaining engagement, thus this study aimed at investigating if there are factors that impact student attention to hints and that could be leveraged by a student model to make these hints more effective. Offline statistical analysis of the gaze data collected from 12 students (age 10–11) playing Prime Climb showed that attention to hints is significantly affected by the following factors: *time of hint* (i.e., whether a hint is given in the first or second half of a Prime Climb session), *hint type*, *attitude toward receiving help* (i.e., whether a student likes receiving help or prefers to do things without help), *game action correctness* and *pre-test scores* (i.e., how much factorization knowledge the student has before starting to play the game). Thus, this offline analysis indicates that capturing these factors and student attention to hints in the Prime Climb student model could help tailor hint presentation to a specific student. Muir and Conati (2012) also found that increased attention to hints was significantly correlated to increased correctness of the subsequent action, showing that further investigation on how to increase student attention to hints is a worthwhile endeavor, because it can improve student performance with the game, and possibly, help trigger student learning.

Eye-tracking has also been used to investigate the students’ interaction and usage of OLMs. Since the mid-1990s, OLMs have attracted a lot of attention within the research community. Allowing the student to access an abstraction of the student model is beneficial in several ways. First, by opening the student model, ITSs become more user-friendly. Many projects have shown that students are capable of scrutinizing their models in order to explore the adaptive nature of the systems, and are interested in seeing the OLMs (Bull et al., 2005; Bull et al., 2007). Moreover, students can be actively involved in the modeling process via OLMs, as some systems allow students to challenge or even update their own student model. Finally, OLMs encourage students to think about their own knowledge, thus involving the student at the meta-cognitive level.

Eye-tracking has been used in several projects to investigate how students process the information in OLMs and evaluate the effectiveness of various types of OLMs. Bull, Cooke, and Mabbott (2007) investigate students’ exploration of six different OLMs for the domain of C programming: a ranked list of concepts, a textual summary of the student model, a hierarchical lecture structure, a concept hierarchy, prerequisite relationships between concepts, and a concept map. In all views except the text summary, color is used to indicate knowledge level, with shades of green indicating correct understanding, yellow and white indicating low knowledge, and red indicating misconceptions. The participants were asked to interact with the OLMs, edit them, and/or persuade the system to change student models. The eye-tracking sessions lasted for 10 min, and students’ preferences for various OLM views were collected via a user questionnaire. Participants generally found the OLMs useful, but had different preferences for which OLMs to use, and spent more time viewing misconceptions in their preferred views. Participants spent much more time examining their knowledge level (which promotes reflection) using the textual representation and ranked concept in comparison to the concept map and the prerequisites. The more complicated OLM views resulted in a broader spread of attention; for example, in the concept map participants focused less on their knowledge level but instead examined the map itself (i.e., they focused on the concepts for which there were insufficient data in their student models). Such more complicated OLMs require more effort from the student to gain an overview of the relationships between concepts.

Mathews et al. (2012) also used gaze data to analyze how students interpret OLMs in the context of EER-Tutor, a constraint-based ITS that teaches conceptual database design. The participants of the study were familiar with EER-Tutor, having used it previously in a database course. The participants viewed four different OLM views: concept tag cloud, kiviat graph, concept hierarchy, and tree map. The goal of the study was to see whether the students understood the OLMs they were presented. The participants were asked three questions about each of the OLM views. For example, participants were asked how much the student (represented by a provided OLM view) had learnt about a particular concept. To answer questions, participants needed to examine the provided OLM. The eye-tracking data were collected in addition to the answers provided by participants. The efficiency of an OLM view was calculated as the quotient of the participant's score (on the answers) and the product of the time spent viewing the OLM view and the number of fixations. A significant difference was found between the efficiencies of the four OLM views. Kiviat graphs and concept hierarchies were significantly easier to interpret in comparison to tag clouds and tree map according to the efficiency measure. Responses from the user questionnaire also identified tag clouds and tree maps as difficult to use to answer precise questions about knowledge levels. Participants were asked to rank the four OLM views by their preference: the highest ranked OLM was kiviat graph, followed by tag clouds, concept hierarchy, and finally, tree maps. Participants commented that the kiviat graph was best for an overall understanding of the student's knowledge, but that the concept hierarchy was valuable for more comprehensive understanding.

As the last chapter in this section, we report work indicating that an additional type of eye-based data, namely, pupillary response, can be leveraged for offline analysis of relevant student states during interaction with an ITS. Muldner et al. (2009) looked at the relationship between pupil dilation and relevant student affective and meta-cognitive states during interaction with EA-Coach, an ITS that helps students learn from analogical problem solving by scaffolding the relevant meta-cognitive skills of *self-explanation* and *analogical reasoning.* A study was conducted with 15 university students who verbalized their reasoning and affective states while interacting with the EA-Coach. The collected protocols were coded for meta-cognitive events (e.g., student utterances indicating *self-explanation*, *analogical reasoning*, and *other forms of reasoning* not falling into the first two categories), as well as for *valence* of affective states (i.e., negative vs. positive affect). The data analysis revealed that type of meta-cognitive event significantly affects pupillary response, with pupil size being statistically significantly larger for *self-explanation* events than for *other forms of reasoning*. Affective valence also had a significant effect on pupillary response, with pupil size being statistically significantly smaller during expressions of negative affect than during expressions of positive affect. The analysis in Muldner et al. (2009) does not provide concrete suggestions on how pupillary response can be used in real time for detection of positive versus negative affect or different types of meta-cognitive events. However, the fact that an effect of these states on pupillary response was found indicates that pupillary response should be further investigated as an additional source of information for student modeling.

Recommendations and Future Research

In this chapter, we have discussed existing research relevant to understand the value of eye-tracking data in student modeling for ITS. This research indicates that the potential of eye-tracking data for student modeling is substantial, because there is evidence that these data can provide information on relevant learner states at the behavioral, cognitive, meta-cognitive, and affective level. In particular, work by Gluck et al (2000), Conati and Merten (2007), Kardan and Conati (2013), and Bondareva et al. (2013) show explicitly that a learner's eyes sometimes reveal more about cognitive and meta-cognitive processes than "overt actions" in a tutor interface. It follows that eye-tracking has the potential to enrich "standard" learner modeling techniques (i.e., those tapping only the regular interaction data).

Further research, however, is necessary to uncover the full extent of this potential. Eye-tracking data have so far been used to direct the adaptive behavior of an ITS by capturing only simple gaze patterns indicating attention or lack thereof (e.g., Silbert et al., 2000; Anderson, 2002; Wang et al., 2006; D'Mello et al., 2012). Student models that leverage gaze data to capture higher level student states such as learning (Kardan and Conati, 2012; Kardan and Conati, 2013, Bondareva et al., 2013), meta-cognition in terms of self-explanation (Conati and Merten, 2007), and affect in term of motivation (Qu and Johnson, 2005) have been developed, validated in terms of accuracy, but not integrated in an ITS. Although it is encouraging that positive results in terms of ITS pedagogical effectiveness have already been obtained by relying on simple gaze patterns (D'Mello et al., 2012), the next step for research in this area will be to see if and how ITS effectiveness can be improved by relying on more sophisticated gaze-enhanced student models.

Another relevant next step is to exploit some of the insights generated by research on offline analysis of gaze data described in this chapter, to extend the usage of gaze data in student modeling and ITSs. For instance, although the findings of Gluck et al. (2000) on lack of attention to an ITS's interventions were exploited in Anderson (2002) to devise an ITS that can track this lack of attention and react to it, the work of Muir and Conati (2012) on factors that affect attention to hints can be leveraged to further improve how an ITS can increase this attention. For instance, Muir and Conati found that *attitude toward receiving help* generates consistent patterns of attention to hints throughout the interaction with the Prime Climb edu-game (low attention for those who do not want help, higher attention for those who do). Thus, if a student model can "see" that the student is not attending to a hint and knows that the student has a negative attitude towards receiving help, it can employ strategies specifically designed to increase attention to hints in someone who does not like receiving help, as opposed to using generic prompts as in Anderson (2002). It would also be interesting to investigate if and how the results uncovered by Muir and Conati (2012) generalize to other types to edu-games and to ITSs at large, and whether other factors may affect attention to hints (e.g., affective state, cognitive overload). A similar analysis could also be done for gaining more detailed insights on which factors affect attention to OLMs in general, and to specific ways to visualize them, especially considering recent results on the impact of individual differences (e.g., perceptual abilities and visualization expertise) and on visualization effectiveness (e.g., Conati and Maclaren, 2009; Toker et al., 2012). Finally, the results in Muldner et al., (2009) indicate that further research should be devoted to investigating how to use information on pupil dilation in student modeling.

Given that research on eye-tracking and student modeling is at a very early stage, as demonstrated by the relatively short list of references at the end of this chapter (we included all relevant articles we could find), should authoring tools for ITSs, such as ASPIRE, CTAT, and GIFT, support the use of eye-tracking data? If so, how? The answer to these questions depends on whether one views the primary purpose of such tools to support ITS research or support development of deployment-ready systems. Both are legitimate purposes and truly versatile authoring tools would cover both. Given that resources for development are always limited, however, existing tools tend to be more oriented towards either one purpose or the other.

When supporting ITS research is a priority, supporting the use of eye-tracking data would be an interesting forward-looking feature for an ITS authoring tool such as GIFT. Given that no best practices for employing eye-tracking data in student modeling have emerged yet, the authoring tool should support rapid prototyping of different ways of building student models that leverage gaze data. This capability would be of tremendous help in studying how eye-tracking might enhance student modeling. A useful first step is to enable researchers to do offline analysis of eye-tracking data combined with other key data sources, such as tutor log data. At minimum, this would require syncing the different data streams so they share common time stamps. A good next step would be to create a versatile architecture that enables the student modeling module (and perhaps other key modules of the ITS) to have access, at run time, to data from an eye-tracker. Steichen et al. (2013) have recently completed an eye-gaze service architecture to address exactly this need for data at run time. Their system, called Eye Movement Data Analysis Toolkit in Real Time (EMDAT-RT), is a standalone application that can provide real-time eye-gaze statistics to third-party applications through a lightweight web service interface. A client application (e.g., an ITS) can simply place a request for eye-gaze analysis (either at regular intervals or specific times), to which the service responds with real-time statistics (calculated either starting from a specific start time or for a specific time window, e.g., the last 10 seconds). Their system integrates a feature-rich open-source eye-gaze analysis module (called EMDAT), capable of calculating numerous summative gaze statistics beyond those usually provided by the analysis packages that come with commercial eye-trackers. The application has been designed to be application-independent, and may therefore be reused for different application domains and purposes, including ITSs. The system (EMDAT-RT) and the internal analysis module (EMDAT) are currently compatible with Tobii eye trackers and will be released as open-source packages soon. Since both offline and online processing require interfacing with an eye-tracker's low-level API, an important goal would also be to make these tools independent of specific eye-tracker models or manufacturers, to increase versatility.

References

Anderson, J. R. (2002). Spanning seven orders of magnitude: A challenge for cognitive modeling. Cognitive Science, 26(1), 85-112. doi:10.1207/s15516709cog2601_

Azevedo, R., Behnagh, R., Duffy, M., Harley, J., Trevors, G.: Metacognition and self-regulated learning in student-centered leaning environments. Theoretical foundations of student-centered learning environments (2nd ed.). 171–197 (2012).

Bondareva, D., Conati, C., Feyzi-Behnagh, R., Harley, J., Azevedo R., Bouchet, F. (2013). Inferring Learning from Gaze Data during Interaction with an Environment to Support Self-Regulated Learning. In Proceedings of AIED 2013, 16th International Conference on AI in Education, Springer.

Bull, S., Cooke, N. & Mabbott, A. (2007) Visual attention in open learner model presentations: an eye-tracking investigation. Proceedings of UM 2007, Eleventh International Conference on User Modeling, (pp. 187-196.), Springer.

Conati C. & Maclaren H. (2008). Exploring the Role of Individual Differences in Information Visualization. Proceeding of AVI 2008, International Working Conference on Advanced Visual Interfaces, ACM Press.

Conati C. & Merten C. (2007). Eye-Tracking for User Modeling in Exploratory Learning Environments: an Empirical Evaluation. Knowledge Based Systems , 20(6), Elsevier Science Publishers.

D'Mello, S., Olney, A., Williams, C. & Hays, P. (2012). Gaze tutor: A gaze-reactive intelligent tutoring system. International Journal Human-Computer Studies, 70(5), 370-389. Elsevier.

Gluck, K. A., Anderson, J. R. & Douglass, S. (2000). Broader bandwidth in student modeling: What if ITS were "eye"TS? In Proceedings of the 5th international conference on intelligent tutoring systems (ITS 2000) (pp. 504-513). Springer-Verlag

Johnson, W.L., Rickel, J.W. & Lester, J.C. (2000) Animated pedagogical agents: face-to-face interaction in interactive learning environments. Int. Journal on Artificial Intelligence in Education, 11, 47-78.

Kardan, S. & Conati, C. (2012). Exploring Gaze Data for Determining User Learning with an Interactive Simulation. Proceedings of UMAP 2012, the 20th International Conference on User Modeling, Adaptation, and Personalization, (pp. 126-138). Springer LNCS 7379.

Kardan, S. & Conati, C. (2013). Comparing and Combining Eye Gaze and Interface Actions for Determining User Learning with an Interactive Simulation. Proceedings of UMAP 2013, the 21th International Conference on User Modeling, Adaptation, and Personalization, Springer.

Mathews, M., Mitrovic, A., Lin, B., Holland, J. & Churcher, N. (2012) Do your eyes give it away? Using eye-tracking data to understand students' attitudes towards open student model representations. In Proceedings of ITS 2012, 11th International Conference on Intelligent Tutoring Systems, (pp. 424-429). Springer-Verlag, LCNS 7315.

Muir, M. & Conati, C. (2012). An Analysis of Attention to Student-Adaptive Hints in an Educational Game. Proceedings of ITS 2012, 11th International Conference of Intelligent Tutoring Systems (pp. 112-122) Springer LNCS 7315.

Muldner, K., Christopherson, R., Atkinson, R., Burleson, W. (2009) Investigating the Utility of Eye-Tracking Information on Affect and Reasoning for User Modeling. Proceedings of UMAP 2009, 17th International Conference on User Modeling, Adaptation, and Personalization, (pp. 138–149). Springer-Verlag.

Qu, L. & Johnson, L. W. (2005). Detecting the Learner's Motivational States in An Interactive Learning Environment. In Proceedings of AIED 2005, International Conference on AI in Education, (pp. 547 – 554), IOS Press.

Sibert, L., Gokturk, M. & Lavine, R. (2000): The reading assistant: eye gaze triggered auditory prompting for reading remediation. Proceedings of UIST 2000, 13th Annual ACM Symposium on User Interface Software and Technology (pp. 101-107). ACM Press.

Steichen, B., Schmid, O., Conati, C. & Carenini, G. Seeing how you’re looking - Using Real-Time Eye Gaze Data for User-Adaptive Visualization. Submitted to 1st International Workshop on User-Adaptive Visualization (WUAV 2013), in conjunction with the 21st Conference on User Modeling, Adaptation and Personalization (UMAP 2013).

Toker, D., Conati, C., Carenini, G. & Haraty, M. (2012). Towards Adaptive Information Visualization: On the Influence of User Characteristics. Proceedings of UMAP 2012, the 20th International Conference on User Modeling, Adaptation, and Personalization (p 274-285). Springer LNCS 7379.

Wang, H., Chignell, M. & Ishizuka, M. (2006) Empathic tutoring software agents using real-time eye-tracking. Proceeding of ETRA 2006, Symposium on eye-tracking research and applications (pp. 73-78). ACM Press.

CHAPTER 22 –Shared Mental Models of Cognition for Intelligent Tutoring of Teams

J.D. Fletcher[1] and Robert A. Sottilare[2]
[1]Institute for Defense Analyses - Science & Technology Division,
[2]U.S. Army Research Laboratory - Human Research and Engineering Directorate

Introduction

This chapter discusses ways ITS principles and structures found in GIFT (Sottilare, Brawner, Goldberg, and Holden, 2012) might be applied to the training of teams. It concerns using the cognitive models of team purpose, behavior, and functions that are shared – held in common – by individual team members to training for teams in a manner analogous to the use of cognitive models in ITS for individuals. We recognize that non-cognitive factors (e.g., physiological and affective) influence team performance and processes. For this chapter, however, we have chosen to focus on cognitive factors.

The use of fully automated, computer-based tutoring technologies to provide training for teams is as embryonic as the problem space is complex. A necessary step in determining optimal strategies for team learning is to assess the collective state of the team, which may be accomplished through the use of shared mental models. Empirical evidence suggests that these models, contribute substantially to successful team performance (e.g., Cannon-Bowers, Salas & Converse, 1993; Rentsch & Hall,1994; Stout, Cannon-Bowers, Salas & Milanovich, 1999; Salas & Fiore, 2004; Banks & Millward, 2007; DeChurch & Mesmer-Megnus, 2010; Espevik, Johnsen & Eid, 2011). However, the notion that shared mental models of cognition within teams might somehow be additive or averaged among team members appears untenable. If AI tutoring is to equal or perhaps exceed skilled human tutoring, success will demand more elegant and powerful approaches for assessing these models and the learning state of teams. These approaches must accurately sense and interpret the critical individual behaviors, team interactions, and environmental factors that promote/inhibit team performance.

Shared mental models represent team objectives and the actions, both individual and collective, needed to achieve them. These models represent team communication and coordination, team posture, situation, and environment, and team member roles and responsibilities. They enable team members "to interpret cues in a similar manner, make compatible decisions, and take appropriate action" (Cannon-Bowers & Salas, 2001, p. 196). Their application in the design and development of intelligent training capabilities for teams appears to be a natural and promising approach for consideration.

The motivation for developing/maintaining shared mental models of cognition is much the same as for maintaining individual models of cognition. For individuals, we refer to the adaptive tutoring learning effect chain (Figure 22-1), where selective mining of learner data (e.g., behaviors and sensor inputs) informs learner states (e.g., cognition, affect), which informs strategy and tactics selection by the tutor and ultimately influences learning gains. Better models of learner cognition result in accurate strategy selection and in improved learning (e.g., knowledge acquisition, skill acquisition).

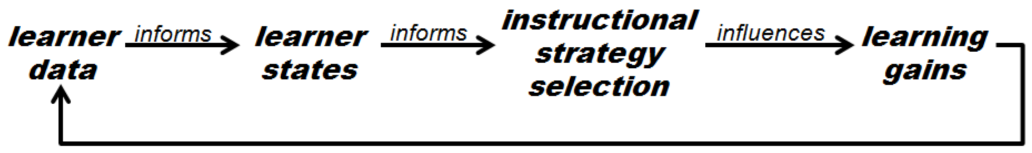


Figure 22-1. Adaptive tutoring learning effect chain (Sottilare, 2012)

When learners train as part of a group, they can encourage each other to ask questions, explain or justify their opinions and reasoning, and actively reflect on their developing knowledge and team performance. Research has shown these activities enhance group performance and individual learning outcomes (especially motivation and engagement – Tchounikine, Rummel, and McLaren, 2010). However, these benefits can only be achieved in well-functioning, actively learning teams (Jarboe, 1996; Soller, 2001). While some teams may demonstrate successful interaction and communication naturally, others may be incapable of developing a balance of participation, leadership, understanding, and encouragement (Soller, 2001). This inability can rapidly degrade group and individual performance, motivation, and engagement, and thereby learning.

As Figure 22-2 suggests an adaptive tutoring learning effect chain model could be extended for teams and then specifically adapted to focus on shared mental models of cognition.

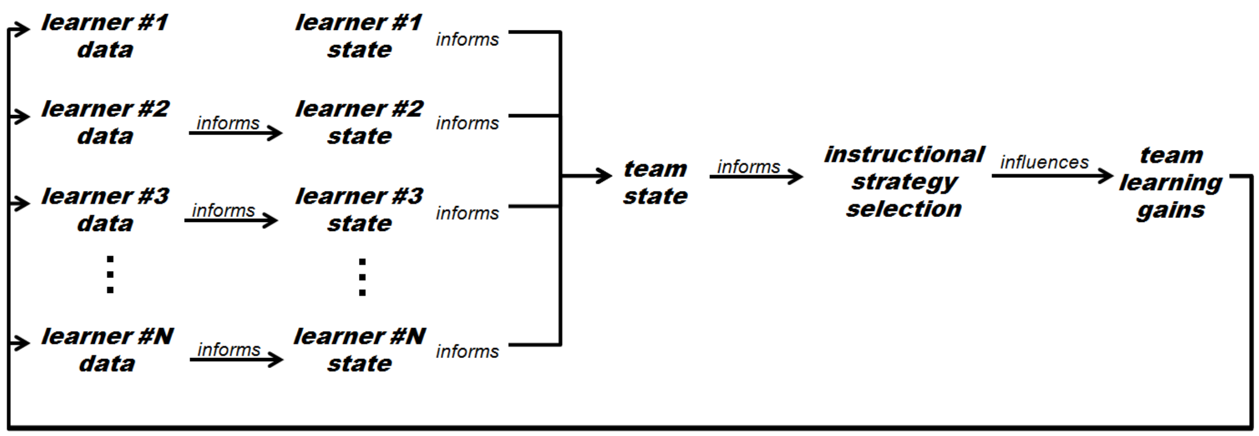


Figure 22-2. Notional Adaptive Tutoring Learning Effect Chain for Teams

Mental Models – Shared and Otherwise

Rouse and Morris (1986) identified common themes in the use of mental models. They described these models as "mechanisms whereby humans are able to generate descriptions of system purpose and form, explanations of system functioning and observed system states, and predictions (or expectations) of future system states" (p 351). Mental models are often dynamic – acting as mental simulations.

Shared mental models may then be viewed as descriptions, explanations, and predictions that the members of a group, such as a team, hold in common. In the case of teams, they are defined by Cannon-Bowers, Salas, and Converse (1993) as "*knowledge structures held by members of a team that enable them to form accurate explanations and expectations for the task, and in turn, to coordinate their actions and adapt their behavior to demands of the task and other team members*" (p. 228).

Research on mental models intensified in the mid-1960s with the evolution of general theories of perception and learning. These theories evolved from the fairly strict logical positivism of behavioral psychology, which emphasizes the study of directly observable and directly measurable actions, to what researchers began to call cognitive psychology. Cognitive psychology gives more consideration to internal, less observable processes, which are assumed to mediate and enable human learning and thereby produce the directly observable behavior that is the subject of behaviorist theories.

Cognitive psychology opened the door to consideration of mental models, but its theoretical and empirical foundations preceded it. The notion of mental models may be found in the primordial origins of scientific psychology. For instance, William James (1890/1950) stated his General Law of Perception as the following: "*Whilst part of what we perceive comes through our senses from the object before us, another part (and it may be the larger part) always comes out of our mind*" (p. 747, 1890/1950). A mental model, then, is a mental representation of the perceived world informed, however imperfectly, by our senses.

Despite the early enthusiasm for behaviorism in experimental psychology, empirical support for a more cognitive view continued to grow. In 1967 Ulric Neisser could point to a large body of empirical evidence indicating that many aspects of human behavior, such as seeing and hearing, could not be accounted for solely by external physical cues. His central assertion was "*that seeing, hearing, and remembering are all acts of construction, which may make more or less use of stimulus information depending on circumstances*" (p.10).

Neisser's contribution helped free researchers to pursue new, more "constructivist" approaches to perception, memory, learning, and cognition by emphasizing their necessity. These approaches require an active synthesis of the environment based on a runnable cognitive model – a cognitive simulation – that is validated or modified by cues impinging on the senses. These actively evolving simulations, not the external stimuli alone, are now assumed to account for what an individual understands about the environment.

We can extend these notions to the functioning of teams. As information and data become available to teams, they are not assumed to be taken in "neat." Instead, they appear to be absorbed and integrated into a rapidly evolving collective simulation of the external environment. Team decisions then result from shared cognitive simulations that are run forward under various scenarios and parameters in order to determine optimized courses of action. Team members must therefore take responsibility for the correctness of their own models and for the ability of others to share them.

Determining how teams develop these models and/or simulations and then share their results should considerably strengthen our procedures for assessing team decision making and performance. Creative, accurate, and comprehensive mental models that take account of all salient cues and filter out others of less immediate importance appear to be critical. Rapid decision making that quickly assesses situations and selects among different decision choices may be at a premium. A large, collective working memory seems especially important for tactical teams whose performance depends on the number of cues they process rapidly and accurately.

Teams and Teamwork

As summarized by Salas and Cannon-Bowers (2000), teams may be described as groups consisting of two or more individuals who must interact with one another in order to accomplish a common task, objective, or mission. Roles and responsibilities of individual team members may be specifically assigned, or they may arise spontaneously, depending on team size, team leadership, and presence of newcomers (Guimera, Uzzi, Spiro & Amaral, 2005). These assignments include requirements for communication and coordinated action – absent such requirements these collectives could be groups but not teams. There are, of course, teams within teams – most teams are components of a larger enterprise.

Teamwork differs in the quantity and quality of communication and coordination required. For instance, an early study by Jones (1974) compared baseball, tennis, football, and basketball teams by regressing the effectiveness of individual team members onto team effectiveness and success. Jones found success to be positively associated with the effectiveness of individual members of baseball, tennis, and football teams, but not basketball teams, where success depends on more closely balanced communication, timing, and coordination among members than the other three. The greater the need for these functions, the greater the need to deal with the team as a learning unit – as a learner with its own team mental model – and a consequent greater need to develop and assess shared mental models.

The scope and character of models that team members must share, therefore, differ with team objectives, extent of teamwork required, and roles that team-members play. At some level, however, all team members and their models must share a common understanding of team processes, interactions, and objectives. The extent to which they do and whether or not it matters can be assessed by team success in performing tasks, objectives, and missions.

Training for teams must adapt to or even prepare for the self-organization and self-assembly that occur in all teams (Guimera, Uzzi, Spiro & Amaral, 2005). This preparation seems especially important for the pick-up teams that are frequently and inevitably assembled to perform military operations. Such teams initially lack the “transactive memory” developed by members of established teams. This memory contains the knowledge and skills of specific team members and an awareness of who can perform team tasks under what conditions of motivation and support (Wegner, 1986). It allows for a division of cognitive labor within a team, permitting the team’s collective knowledge to exceed that of any individual team member.

Studies reviewed by Lewis and Herndon (2011) found a strong positive relationship between transactive memory and team performance. One reason for this finding may be the long noted inverse relationship between frequency of communication and the quality of team performance in reviews of collective behavior (Briggs & Johnston, 1967; Olmstead, 1992). Communications can be minimized only if the members of teams share a common understanding of the situation and what can be done by whom.

Intelligent Tutoring Systems

The definition of an ITS varies across researchers, designers, and developers and is discussed elsewhere in this volume. In accord with GIFT, an ITS may be viewed as an effort to capture in computer technology the capabilities and practices of a human instructor who is expert in both the subject matter and one-on-one tutoring.

ITS development is motivated by the empirically evident benefits of human tutoring (e.g., Bloom, 1984; Graesser, D’Mello & Cade, 2011; VanLehn, 2011) and a long-standing desire to make these benefits more widely accessible and affordable than those delivered by human tutors (Fletcher, 1992, 2009; Corbett, 2001). Another motivation for the development of ITSs grew from the recognition that although computers could be used to teach effectively, it took time and considerable expense to anticipate all possible states of the learner and program all possible instructional responses to these states. Response to both of these motivations requires a generative capability, which is a defining characteristic of ITS. Dynamic information structures and mixed-initiative in computer-based tutorial dialogue were intended to generate instructional interactions in real time, thereby relieving much of the burden and cost of authoring adaptive, individualizing instruction (Carbonell, 1970; Fletcher, 2009; Fletcher & Rockway, 1986). To an appreciable extent, an ITS should eventually become a self-authoring system. With the capability to access almost all human knowledge through the global information infrastructure, ITS capabilities may make learning affordable and universally accessible, generated on demand – anytime and anywhere (Fletcher, 2006, 2009).

As in most technologies, ITS development begins with a metaphor, i.e., producing computer systems that clone human tutors. Just as wireless telegraph led to radios, horseless carriages led to automobiles, and so on, tutor-less tutoring may evolve into something as yet unforeseen. Sooner or later the “Columbus Effect” will exert its influence, but, for current ITS development and this chapter, this metaphor may suffice.

ITS capabilities were early envisioned by Uttal (1962), Feurzeig (1969), and Carbonell (1970). They have been pursued into the present. Today ITS development suggests a future in which education, training, and performance aiding do not take place solely through prefabricated lessons and other material but are accomplished in the form of one-on-one, guided dialogues, that are generated on demand, tailored to the needs, abilities, interests, and values of individual learners, and are based on mixed-initiative conversations in which either the computer/tutor or the learner may take the initiative. Although not widely found, dialogues of this sort have been available since the 1970s (e.g., Brown, Burton, and DeKleer, 1982). Eventually, they may provide a Plato for every Aristotle, an Aristotle for every Alexander, and a Mark Hopkins for the rest of us.

ITSs can be contrasted with drill and practice programs. The latter methods were found to be very effective in achieving lower level instructional objectives such as learning arithmetic facts (Suppes & Morningstar, 1972), grapheme-phoneme correspondences in beginning reading (Fletcher & Atkinson, 1972), and foreign language vocabulary and phonetics (Van Campen, 1981).

Such rudimentary objectives are found in initial learning of practically all subject domains. They consist of discrete items, simple concepts, or straightforward procedures to be memorized and/or applied and are limited to objectives in the lower reaches of Bloom’s (1956) hierarchy or the lower left-hand corner of Anderson and Krathwohl’s (2001) learning space. Drill and practice programs have a strong role to play at this level. They are effective and inexpensive to design, develop, and deliver (Fletcher, 2006). They require models of the learner, but all relevant states of the learner must be anticipated at design time and pre-programmed into the system. Learner modeling in these systems is predominately pre-assigned, implicit, and static. As effective as drill and practice programs are for helping learners master domain rudiments, they are limited in getting beyond these.

ITSs are not unique in their use of learner models, but their approach to learner modeling is fundamentally different from drill and practice. ITS learner models are dynamic and generated on demand as needed by the instructional program. They are explicit, often with reference to comprehensive models of both the procedures and knowledge required to successfully attain instructional objectives. Because of their dynamic qualities, they are particularly suited to tutorial dialogue systems that must generate instructional and problem solving guidance on demand, in real time.

Development of ITSs can aim for more conceptual, abstract, and analytical objectives, where their capabilities are better used, the expense to develop them is better justified, and they are evidently more effective (cf. Feurzeig, 1969). Effect sizes from ITS studies by Grasser, et al. (2003), Person, et al. (2001), and VanLehn et al. (2005) average about 0.62 for deep learning compared to –0.02 for shallow learning (Kulik & Fletcher, 2012).

As suggested above, the subject domain rudiments needed for teamwork can be provided efficiently and effectively through individual drill and practice. Notably, much collective training of teams is provided through practice and feedback – very much in a drill and practice manner. As in individual training, team training objectives need to transcend subject domain rudiments as, for instance, Salas and Cannon-Bowers (2000) discuss in detail. To do so, requires the ability, found in ITSs, to deal with higher order team capabilities.

A specific strength of ITS is based on their generative capabilities to identify and then provide instruction that deals with unanticipated learner states of individuals as seen in the knowledge and model tracing discussed by Anderson, Boyle, Corbett, and Lewis (1990) and Anderson, Corbett, Koedinger, and Pelletier (1995). Anticipating possible learner states for teams with their varying membership, the evolving roles and responsibilities of team members, and transactive understanding of team communication and coordination, is likely to be exorbitant in both cost and time – if not impossible. This problem is partially finessed by after action reviews (e.g., Morrison & Meliza, 1999), but these occur after the fact and, despite instrumentation, are dependent on subjective impressions and recollections. ITS techniques may have much to contribute in modeling team states and applying their capabilities to develop, and adjust team training, possibly in real time. The GIFT framework may well be used to examine this possibility (Sottilare et al., 2012).

GIFT's service-oriented ITS architecture and methodology may be summarized as containing four major components:

- An interactive interface (for mixed-initiative dialogue, allowing either the learner or the computer-based tutor to initiate queries and discussion);
- An explicit model representing the knowledge and skills that form the objectives of the instruction (where we want to go);
- An explicit dynamic model of the individual learner's evolving knowledge, skills, and progress toward achieving the objectives of the instruction (where we are now); and
- Tutoring strategies that use these models to bridge the gap between the learner's current knowledge and skills and the targeted instructional objectives (getting from here to there).

These components will be familiar to most developers of ITS. GIFT's contribution is in the details of their development, its modular and service-oriented architecture, and the particular attention it pays to sensors by separating out a sensor module as a major component of its framework.

Team Training, Shared Mental Models, and Intelligent Tutoring Systems

An issue at hand is whether team training can be informed by what we have learned about developing computer tutors for individuals. This consideration suggests two obvious levels of learning. The first level concerns the knowledge and skills of individual team members. At this level and as briefly described above, a highly effective learning environment is created by one tutor working with one learner. Meta-analytic reviews by VanLehn (2011) and Kulik and Fletcher (2012) found this approach to be substantially more effective than classroom instruction where opportunities for individualized, tutorial instruction are limited. Instructional technology has made tutoring not only accessible but also affordable by imbuing computers with the capabilities of human tutors.

To an appreciable extent, team performance is a product of personnel selection. Individuals chosen for a team need to possess the levels of knowledge and skills required by their team roles and responsibilities and to complement the strengths and weaknesses of other team members. Much training for team membership may be accomplished by individual training to meet the standards and conditions of performance required by an occupational specialty and the level of skill sought within it. The capability and likely performance of a team could even be viewed as nothing more than the sum of the competencies provided by the individual training received by its members. This might be true if teams were not composed of people.

People bring a notoriously wide range of individual differences to teams. These differences are found even when people who possess formally identified competencies can be identified, located, and assigned, which, itself, is not always the case. Beyond cognitive differences, people also bring to teams different attitudes, motivations, interests, and values. These differences strongly affect a team's abilities to perform its missions. Researchers have repeatedly and empirically found that team training and team cognition transcend the sum of individual training received by team members (e.g., Stout, Salas & Carson, 1994; Liang, Moreland & Argote, 1995; Salas & Fiore, 2004; DeChurch & Mesmer-Magnus, 2010).

A second level is needed, then, to train the team as a coherent collective. Although it relies on the prior individual training of team members, collective team training remains critical to the success of nearly all teams – and particularly teams formed to carry out military operations. Thompson's (1967) hierarchy of pooled, sequential, and reciprocal interaction and Van de Ven, Delbecq, and Koenig's (1976) methods of exchange within teams suggest increasing levels of interdependence of individual team members and increasing need for coordination among those members, and thus providing insight into what the ITS must know about the type of tasks being trained.

In pooled team models (e.g., a team of painters painting rooms in a house) where there is low task interdependence, the workload of a tutor is lower. The tutor can simply track each team member's performance and sum them all to determine the team's performance (e.g., total number of rooms painted) at any given time during training. Pooled team members generally have the same skills and roles.

For sequential team models (e.g., running a relay race) where task interdependence is higher since one member must complete an action before the next one begins, the tutor can track the output/performance of the last team member in the sequence to determine the team's overall performance, and track individual performances earlier in the sequence to project overall performance. Individual team members in sequential tasks may (e.g., relay race runners) or may not (e.g., assembly line workers) have similar roles.

The ITS workload quickly ramps up as the directionality of the workflow increases. Reciprocal, or two-way workflow, means that each team member can be both a source and a recipient in the workflow. Since reciprocal team members tend to have specialized roles, workflow and thereby performance can be compromised by subtasks with longer duration than other subtasks. For example, subtask A takes team member A an average of five minutes to complete while subtasks, B and C take two and three minutes for team members B and C to complete. Assuming that the subtasks could be done in any order, team members B and C are more likely to have downtime waiting for team member A to complete a subtask. The ITS must be aware of the characteristics of the reciprocal workflow and subtasks to avoid providing feedback unnecessarily and negatively influencing the quality of the work.

The complexity increases again as directionality increases from two (reciprocal) to a multi-directional (team model). To overcome this complexity and increase the probability of success, it would be useful to have teams with members who have multiple specialties and can switch tasks during downtimes. This is not always possible. The object of training is to build new knowledge and skills. The ITS' "understanding" of the problem space and complexity is essential in developing the individuals on the team and enhancing the performance of the team. Similar to individual tutoring, team tutoring relies on a recognition of when the team is at expectation, below expectation, or above expectation.

Might ITS capabilities developed for a single tutor working with a single learner apply to multi-learner teams? Might they be applied to develop team competencies, knowledge, cognition, and performance? For teams, and in accord with Sottilare et al. (2012), these questions may be structured around GIFT modules for sensors, learners, pedagogy, and domains. In today's GIFT, modules include models and software processes to manage data, turn into information (e.g., states), and then use that information to make decisions about instruction for individuals. To extend GIFT for use with teams, we examine the existing modules (sensor, learner, pedagogical, domain) and recommend enhancements to these and rationalize the development of specific team models.

Challenges and Gaps in Developing Shared Mental Models

While there are many challenges in moving forward with team training and the development and use of shared mental models in the process, some appear especially significant. A key to establishing effective collaborative learning is the ability of the tutor to manage uncertainty and dynamic nature of team interaction and communication.

Team members come and go. Few teams work as an established unit with the same members over extended periods of time. The social interaction among team members that is necessary for trust-building will not always foster learning (Brown and Palincsar, 1989). Traditionally, trainees view learning as an independent and mildly competitive activity. Many trainees hesitate to ask for help from their peers for fear of appearing incompetent or dependent. Furthermore, peers tend to work together with the aim of simply accomplishing tasks (e.g., finding the right answers) instead of facilitating each other's learning. The probability that all trainees understand the learning material and progress as a team increases when each member understands the roles and responsibilities, and actively participates in the training process (Soller, 2001). Shared mental models of team confidence and commitment may be essential tools for ITSs to promote active participation; encourage the exchange ideas, information, and perspectives for interaction; provide real-time monitoring of individual and team participation level (e.g., interaction analysis); and manage low participation levels.

Another significant challenge is for the ITS to understand the relationship between team and individual performance and actions. Roles and responsibilities must be defined so the computer-based tutor can aggregate individual actions in a logical, weighted fashion and adapt to the team performance state. The tutor must also understand when to provide feedback to the individual team members based on positive actions (e.g., goals met) or negative actions (e.g., distracting off-task behavior).

Peer interactions (and thereby their associated mental models) may change as the training domain changes. Interactions have been found to vary enormously even within the same domain (Brown and Palincsar, 1989). One aspect that contributes to this uncertainty in trainee communication is ill-defined roles and goals, and the adeptness of the team members at switching roles and between tasks (Burton, 1998). Role identification and switching is good for social grounding and can create an environment for collaborative learning and more effective communication (Soller, Linton, Goodman and Gaimari, 1998). This indicates that the ITS might be more effective if it could develop and maintain a model of team adaptability. As the tasks and objectives become more complex, effective communication within the team becomes more important, and the ability of the team and its members to adapt may lead to a richer learning experience. An ITS should be able diagnose and redirect incorrect solution paths, divide complex tasks into sub-tasks associated with assigned individual team members, clusters of team members, and the entire team. The idea of a team tutoring system as observer, manager (decision maker), and director is evolving.

While individual behaviors are observable, a primary challenge in developing shared mental models for ITSs arises from other, unobservable cues to individual states. Stress and anxiety, which limit cognition, may manifest themselves in outwardly observable behavior by novices, but may be more veiled by experts who have learned to set aside external stressors to focus on the task at hand. This is where physiological sensors have the potential to play a significant role in cognitive state detection. For instance, electro-dermal activity (EDA) has been shown to indicate stress and anxiety (Scheirer, Klein, Fernandez, and Picard, 2002). What is needed is a mechanism to indicate the source of the stress in order for the ITS to help manage the training experience through guidance (e.g., scaffolding) and optimize the difficulty level of the training experience – in accord with Yerkes-Dodson's (1908) inverted U, Vygotsky's (1978) zone of proximal development, or similar notions for adapting difficulty to the learner.

Finally, there is the challenge of training geographically distributed teams and developing their associated mental models. Local teams have been found to learn more than geographically distributed teams (Andres, 2002; Warkentin, Sayeed & Hightower, 1997), and distributed teams exchange information less effectively than local teams (Warkentin, Sayeed & Hightower, 1997). However, with sufficient time to develop strong group relationships and become comfortable with the communication environment, dispersed teams can communicate as effectively as co-located teams (Chidambaram, 1996). The military has developed distributed simulation for training as a way to make all team training affordable and accesible. Mechanisms to develop shared mental models based on the traits and experience of individual team members would be desirable in organizations where teams are short-lived.

Enhancing GIFT Shared Mental Models for Team Training

The GIFT sensor module provides interfaces for behavioral and physiological sensors. It accepts raw sensor data, in some cases processes this data, and then uses this information to determine individual states (e.g., workload, engagement) for transfer to the learner module.

Sensor modules may be at an advantage in team training because many aspects of decision making and problem solving in teams must be carried out explicitly and can be assessed directly, whereas they are implicit and must be inferred for individual training. Collective team cognition and declarative team knowledge remain to be dealt with, but these assessments may also be facilitated by the observable and measurable actions and procedures that are generally the object of team training and the frequently observable and measureable coordination and cooperation required to perform them. Affective team states may be similarly amenable to assessment by sensors, but they and physiological states are set aside for this chapter.

The learner module uses data received from the sensor module, performance and knowledge assessments, and demographic data to determine the learner's present cognitive, affective, and competency states. This state information is sent to the pedagogical module, where states are compared to expected learner states and matched with successful practices of human tutors to determine which instructional strategy should be used next.

Representation of team states and traits in the learner module should, in turn, benefit from sensor data provided by observation of the explicit processes and explicit actions taken in team training. Using sensor module data, a computer-based tutor will be able to deal more accurately and comprehensively in creating a dynamic representation of team cognition in general, the mental models shared by all team members, and the mental models of team communication and coordination being acquired by individual team members. Moreover, the massive amounts of team member historical, demographic, trait data, along with highly granular performance data can be rapidly recorded, accessed, mined, and updated as needed, using machine learning techniques.

The pedagogical module is domain-independent. It uses learner state, performance data, and knowledge models to determine the content, order, and flow of instruction. It recommends general instructional strategies to guide the domain module's choice of domain-dependent tactics. In team training, for instance, the pedagogical module might use ITS techniques for knowledge and performance model tracing to recommend a simulation with the type of scenarios intended to develop a number of strategic, general team capabilities such as adaptability, grit, situation awareness, problem solving, communication, or coordination.

The pedagogical module, in turn, benefits from team member trait and state data compiled by the learner module. These data not only inform decisions about what, pedagogically, to do next with a team, but also how to test the performance predictions on which its prior recommendations were based. Using learner module data, a computer-based tutor will be able to create more accurate, comprehensive, and dynamic representations of team cognition in general, the mental models shared by all team members, and the mental models of all sorts that are being devised by team members and that impact team performance. Additionally, the massive amounts of team member historical and demographic trait data can be combined with highly granular performance data to be rapidly recorded, accessed, mined, and updated as needed to enhance and apply the tutorial capabilities of an ITS.

The domain module is domain-dependent. It defines and structures the instructional domain's declarative and procedural knowledge requirements. It translates the pedagogical module's strategic recommendations into domain-specific instructional tactics, which determine the content, order, pace, and feedback alternatives for presentation to the learner. For instance, it will assess and predict the learner's progress toward achieving instructional objectives.

The domain module will similarly apply ITS capabilities to learn and improve domain-specific instructional tactics as it responds to the strategic recommendations received from the pedagogical module. It will return accurate, real-time feedback to the learner module to help it model the team state, refine its representation of team traits, diagnostically assess the team's progress toward achieving its overall instructional objectives, including those recommended for emphasis by the pedagogical module, and diagnostically model the development of individual team members in fulfilling their team roles and responsibilities.

In short, the GIFT modules and their functions may in many respects perform in team training just as they do in individual training. However, there remain issues that are peculiar to team training and team cognition that require attention in using shared mental models in intelligent training for teams. GIFT's modular, distributed architecture allows for asynchronous interaction and simultaneous tutoring of individuals. This architecture allows for individual feedback in a team context where each learner's tutor communicates changes of individual state to the other tutoring agents within GIFT, thereby supporting both team level models and individual learner models. In other words, the tutor for learner A maintains information about learner A and shares it with the tutor for learners B, C, etc. and the team states (e.g., performance) so each learner has a fully informed tutor.

Sottilare, Holden, Brawner, and Goldberg (2011) considered specific and separate team state models that may be informed by individual learner models and historical team performance data that might be gathered from relevant operational and training environments. In addition to the shared mental models discussed in the previous challenge section, models of performance, competency, cognitive, affective, trust, and communications were also considered. While this chapter has focused on team cognitive factors, it is worth noting that non-cognitive factors, such as affect, morale, confidence, and physical state, which are not discussed here, moderate cognition and their effects should eventually be considered.

Team performance models, as noted above, may consist of observations of team behavior as it progresses toward one or more objectives, including their conditions and standards of performance. Team assessment techniques are crucial in developing a clear understanding of team performance. Salas, Rosen, Held, and Weissmuller (2009) argued that performance measurement works best when it captures and considers performance from multiple sources, it is tightly coupled to the action needed, it uses validated expert models to assess the performance, it directly supports learning, and it provides real-time corrective feedback. Individual and team assessments analyze factors including (1) when each learner is ready to take an action, (2) delay in actually taking that action, and (3) value of the actions taken.

Team competency models may be used to predict performance within a domain. They are based on previous related experiences and associated levels of success. Cumulative team competency models are needed for the ITS to choose initial training scenarios and set expectations for performance. Team learning objectives, individual state information, the interactivity of the training task, and the interdependence of workflow can be used to inform a team cognitive state model that assesses mental workload, engagement and compares progress with expectations to determine team and individual strategies and tactics.

While not specifically targeted in this chapter, modeling of team affect is necessary for optimizing cognitive performance. Behavioral observations may be a pathway to understanding individual and team affect, but more evidence is needed. What is clear is that affect is a moderator of cognition. Problem solving and decision making become more difficult as affect becomes more extreme. A shared mental model of affect can inform the ITS to take action to guide, mediate, or challenge team members to get them back on track.

In accord with Olmstead (1992), among others, the amount and type of communication is considered a significant behavioral indicator of team trust and cooperation for our notional shared mental model. Roles (e.g., leader, follower, domain expert) should also be considered a factor in that roles moderate/indicate communications and expected communications. For example, a team leader would be expected to communicate mission intent (goals), clarify roles, and direct activities as needed. The ITS managing the shared mental model of cognition would be expected to monitor communication to determine how it met or did not meet expectations.

Next Steps

Salas and Cannon-Bowers (2000) chose to emphasize 10 critical questions in their comprehensive chapter on team training. They assert that ITS have great promise for team training and performance measurement, which leads to question 5 in their list, “Can intelligent systems be developed to assess, diagnose, and remediate teamwork?” (p. 331). They go on to focus on dynamic assessment, a real-time assessment capability that provides immediate feedback and automated diagnoses of performances. This thought leads to question 6 in their list, “Can mechanisms of dynamic assessment be developed for teams?” (p. 331).

In applying what we have learned from ITSs to the development of team competencies, an obvious first issue to address is how to extend ITS techniques for modeling an individual learner to modeling teams and team cognition to determine what a team “knows.” Such an extension may do much to inform GIFT learner modules and decisions made by its pedagogical modules, as they are applied to team training,

Much discussion along these lines concerns declarative and procedural knowledge (e.g., Banks & Millward, 2007). Research in the service of ITSs and elsewhere has given us tools for assessing both. For example, ITS designers often use concept models to assess what a learner and teams must know in order to achieve learning objectives (Cooke, Salas, Cannon-Bowers & Stout, 2000). These concept models have tended to bundle declarative and strategic knowledge together -- there may be reasons to separate the two into separate concept models, but that seems best set aside for the moment.

Further, much ITS activity consists of problem-solving exercises in which the progress of learners toward problem solutions can be explicitly and objectively observed (Kulik & Fletcher, 2012). ITS designers employ procedural models that lay out the actions an expert might use to solve a problem in the subject domain. Based on a learner's actions in solving a problem, an ITS can thereby infer what the learner knows. Bayesian techniques, for example, are currently prominent in such inferences. They turn cause and effect on its head, allowing us to estimate the probability of a given cause (e.g., a component of the knowledge model) that brought about the observed effect. These estimates improve as experience with additional learners build up, allowing the learning system itself to learn.

ITS tracing activities (mapping actions taken unto procedure models and inferred knowledge onto knowledge models) lend themselves well to team training, much of which involves exercises and problem solving – practice with feedback. We might apply ITS modeling and model tracing processes to teams in two ways – to identify and assess the declarative (including strategic) knowledge of individual team members and to do the same for the team itself as a collective. Empirical study on both as they apply to the training of teams and accessing their progress toward targeted instructional objectives may do much to develop ITS capabilities for team training. It would answer some long-standing questions.

For instance, in team exercises we have objective data on the performance of individual team members and of the team as a collective. ITS capabilities for inferring the knowledge element could then be used to determine the knowledge models of individuals, the collective knowledge of the team, and how the two compare. They could help determine if there is team knowledge or cognition that is separate from the sum of the mental models of its members, but the nature of this separate, collective model, how it contributes to successful performance of team tasks and missions, and what, if anything, can be done to develop it through training.

A second issue that might be addressed concerns what must be shared among the mental models of team members. Given the research on transactive memory, which is discussed earlier in this chapter, and its evident contributions to successful team performance, it appears that not all team members must possess all team knowledge. This is not a surprising conclusion, but research on team and team member cognition (or mental models) may identify what must be shared, the priorities for sharing whatever separate elements are identified, and, again, what can and should be done about them in team training, which is typically given limited time and resources.

A third issue concerns generic teamwork knowledge and skills that are separate from subject domains and must be acquired by individuals if they are to perform as successful members of a team. Most of this knowledge and skill is acquired in team training environments, which tend to be more expensive and logistically difficult to implement than individual training environments. Significant efficiencies and economies may be realized if at least some knowledge and skill in teamwork can be developed through individual training. That these competencies can be improved through individual training seems likely, but the nature and characteristics of these competencies must be more precisely identified and understood.

A fourth matter concerns the tutorial dialogues that are the eventual target for ITS development. These dialogues seem likely to remain at the individual level, but computer-based tutors could have full access to team exercise instrumentation data, provided by the GIFT sensor module, individual history and other team-relevant information provided by the GIFT learner model, training objectives held in the pedagogical model, and domain-specific data obtainable from the domain module. These dialogues could initially provide private, individualized feedback to team participants. Capabilities to do this are well within the state of the art. Eventually these dialogues might become genuine facilitated discussions with an individual. A research task with fairly rapid return may be to link up ITS dialogue capabilities with team exercise data and provide these as individual feedback. Doing so will extract much more value from training exercises than is now possible because of their accurate and comprehensive access to data and their ability to interact privately with each participant as an individual team member.

Final Thoughts

Other questions, as well as other lines of research, may well occur to readers. As Salas and Cannon-Bowers (2000) suggest, there may be much in the ITS world of value if it is applied to team training. This chapter has focused on mental models and their sharing in team cognition, but many other paths also seem likely to return significant value. Cannon-Bowers and Salas (2001) point out a number of fundamental questions to be addressed by empirical study of shared mental models including determinations of what must be shared, what we mean by sharing, how we should measure it, and what outcomes and value can be expected if we are successful. Our suggestions only begin to fill out the GIFT framework with the specifics needed. Other pathways are available and might well be pursued.

References

Anderson, J. R., Boyle, C. F., Corbett, A. T. & Lewis, M. W. 1990. “Cognitive Modeling and Intelligent Tutoring.” *Artificial Intelligence, 42(1),* 7–49.

Anderson, J. R., Corbett, A. T., Koedinger, K. R. & Pelletier, R. (1995). “Cognitive Tutors: Lessons Learned.” *Journal of the Learning Sciences, 4(2),* 167–207.

Anderson, 1. W. & Krathwohl, D. R. (Eds.) (2001). A Taxonomy for Learning, Teaching, and Assessing: A Taxonomy of Educational Objectives. Columbus, OH: Allyn & Bacon.

Andres, H. (2002). A comparison of face-to-face and virtual software development teams. *Team Performance Management*, 8(1/2), 39-48.

Banks, A.P. & Millward, L.J. (2007). Differentiating knowledge in teams: The effect of shared declarative and procedural knowledge on team performance. *Group Dynamics: Theory, Research & Practice, 11(2),* 95-106.

Bloom, B. S. (1956). Taxonomy of Educational Objectives, Handbook I: The Cognitive Domain. New York: David McKay Co. Inc.

Bloom, B. S. (1984). The 2 sigma problem: The search for methods of group instruction as effective as one-to-one tutoring. *Educational Researcher, 13,* 4-16.

Briggs, G. E., and Johnston, W. A. (1967). *Team Training* (NAVTRADEVCEN 1327-4). Orlando, FL: Navy Training Device Center. (DTIC/NTIS No. ADA 660 019)

Brown, A., and Palincsar, A. (1989). Guided, cooperative learning and individual knowledge acquisition. In L. B. Resnick (Ed.), *Knowing, learning, and instruction: Essays in honor of Robert Glaser* (pp. 393-451). Hillsdale, NJ: Erlbaum.

Brown, J. S., Burton, R. R. & DeKleer, J. (1982). Pedagogical, natural language and knowledge engineering in SOPHIE I, II, and III. In D. Sleeman & J. S. Brown (Eds.), *Intelligent tutoring systems* (pp. 227-282). New York, NY: Academic Press.

Burton, M. (1998*). Computer Modeling of dialogue roles in collaborative learning activities.* (Unpublished doctoral dissertation). Computer-Based Learning Unit, University of Leeds, Leeds, UK.

Cannon-Bowers, J. A. & Salas, E. (2001). Reflections on shared cognition. *Journal of Organizational Behavior, 22*, 195-202.

Cannon-Bowers, J. A. Salas, E. & Converse, S. A. (1990). Cognitive psychology and team training: Shared mental models in complex systems. *Human Factors Society Bulletin, 33(12),* 1-4.

Cannon-Bowers, J. A., Salas, E. & Converse, S. A. (1993). Shared mental models in expert team decision making. In N. J.Castellan, Jr. (Ed.) , *Current issues in individual and group decision making* (pp. 221– 246). Hillsdale, NJ: Erlbaum.

Carbonell, J. R . (1970) AI in CAI: An Artificial Intelligence Approach to Computer-Assisted Instruction. IEEE Transactions on Man-Machine Systems, 11, 190-202.

Chidambaram, L. (1996). Relational development in computer-supported groups. *MIS Quarterly*, 20(2), 143-163.

Cooke, N. J., Salas, E., Cannon-Bowers, J. A. & Stout, R. (2000). Measuring team knowledge. Human Factors, 42, 151– 173.

Corbett, A. (2001). Cognitive computer tutors: Solving the two-sigma problem. In M. Bauer, P. J. Gmytrasiewicz & J. Vassileva (Eds.), *User modeling* (pp. 137–147). Berlin: Springer-Verlag.

DeChurch, L.A. & Mesmer-Magnus, J.R. (2010). The Cognitive Underpinnings of Effective Teamwork: A Meta-Analysis. *Journal of Applied Psychology, 95(1)*, 32-53

Espevik, R., Johnsen, B & Eid, J. (2011). Communication and performance in co-located and distributed teams: An issue of shared mental models of team members? *Military Psychology, 23(6)*, 616-638.

Espevik, R., Johnsen, B.H. & Eid, J. (2011). Outcomes of shared mental models of team members in cross training and high-intensity simulations. *Journal of Cognitive Engineering and Decision Making, 5(4)*, 352-377.

Feurzeig, W. (1969). *Computer Systems for Teaching Complex Concepts* (BBN Report 1742). Cambridge, MA: Bolt Beranek & Newman, Inc. (DTIC AD 684 831)

Fletcher, J.D. (1992). Individualized systems of instruction. In M.C. Alkin (Ed.) *Encyclopedia of Educational Research* (Sixth Edition) (pp. 613-620). New York, NY: Macmillan.

Fletcher, J. D. (2001) What Do Sharable Instructional Objects Have to do with Intelligent Tutoring Systems, and Vice Versa? International Journal of Cognitive Ergonomics, 5, 317-333.

Fletcher, J. D. (2006) The ADL Vision. In, H. F. O'Neill and R. Perez. (Eds.) *Web-Based Learning: Theory, Research and Practice* (pp. 31-53). Mahwah, NJ: Lawrence Erlbaum.

Fletcher, J. D. (2009). Education and Training Technology in the Military. Science, 323, 72-75.

Fletcher, J.D. & Atkinson, R.C. (1972). An evaluation of the Stanford CAI program in initial reading (Grades K through 3). *Journal of Educational Psychology, 63,* 597-602.

Fletcher, J.D. & Rockway, M.R. Computer-based training in the military. In J.A. Ellis (Ed.) Military Contributions to Instructional Technology (pp. 171-222). New York, NY: Praeger Publishers, 1986.

Graesser, A., D'Mello & Cade, W. (2011) Instruction based on tutoring. In R.E. Mayer & P.A. Alexander (Eds.) *Handbook of Research on Learning and Instruction* (pp. 408-426). New York: Routledge.

Graesser, A. C., Moreno, K., Marineau, J., Adcock, A., Olney, A., Person, N. & The Tutoring Research Group. (2003). AutoTutor improves deep learning of computer literacy: Is it the dialog or the talking head. In U. Hoppe, F. Verdejo & J. Kay (Eds.), *Proceedings of Artificial Intelligence in Education* (pp. 47–54). Amsterdam: IOS Press.

Guimera, R., Uzzi, B., Spiro, J. & Amaral, L.A.N. (2005). Team assembly mechanisms determine collaboration network structure and team performance. *Science, 308(5722)*, 697-702.

James, W. (1890/1950). *Principles of Psychology: Volume I*. New York: Dover Press.

Jarboe, S. (1996). Procedures for enhancing group decision making. In B. Hirokawa and M. Poole (Eds.), *Communication and Group Decision Making* (pp. 345-383). Thousand Oaks, CA: Sage Publications.

Jones, M.B. (1974). Regressing group on individual effectiveness. *Organizational Behavior and Human Performance, 11,* 426-451.

Kulik, J.A. & Fletcher, J.D. (2012). *Effectiveness of Intelligent Tutoring Systems* (IDA Document D-4664). Alexandria, VA: Institute for Defense Analyses.

Lewis, K. & Herndon, B. (2011). Transactive memory systems: Current issues and future research directions. *Organization Science, 22(5)*, 1254–1265.

Liang, D.W., Moreland, R. & Argote, L. (1995). Group versus individual training and group performance: The mediating factor of transactive memory. *Personality and Social Psychological Bulletin, 21,* 384-393.

Morrison, J. E. & Meliza, L. L. (1999). *Foundations of the After Action Review Process* (IDA Document D-2332). Alexandria, VA: Institute for Defense Analyses.

Neisser, U. (1967). *Cognitive Psychology*. New York, NY: Appleton, Century, Crofts.

Olmstead, J. A. (1992). *Battle Staff Integration* (IDA Paper P-2560). Alexandria, VA: Institute for Defense Analyses. (DTIC/NTIS No. ADA 248 941)

Person, N. K., Bautista, L., Graesser, A. C., Mathews, E. C. & The Tutoring Research Group. (2001). Evaluating student learning gains in two versions of AutoTutor." In J. D. Moore, C. L. Redfield & W. L. Johnson (Eds.), *Artificial Intelligence in Education: AI-ED in the Wired and Wireless Future* (pp. 286–293). Amsterdam: IOS Press.

Rentsch, J.R. & Hall, R.J. (1994). Members of great teams think alike: A model of team effectiveness and schema similarity among team members (p. . In M.M. Beyerlein & D.A. Johnson (Eds.), *Advances in Interdisciplinary Studies of Work Teams: Vol 1.* Greenwich, CT: JAI Press.

Rouse, W. B. & Morris, N. M. (1986). On looking into the black box: Prospects and limits in the search for mental models. *Psychological Bulletin, 100,* 349– 363.

Salas, E. & Cannon-Bowers, J.A. (2000). The anatomy of team training. In, S. Tobias & J.D. Fletcher (Eds.) *Training and Retraining: A Handbook for Business, Industry, Government, and the Military* (p. 312-335). New York, NY: Macmillan.

Salas, E. & Fiore, S.M. (2004). *Team Cognition: Understanding the Factors that Drive Process and performance.* Washington, DC: American Psychological Association.

Salas, E., Rosen, M. A., Held, J. D., and Weissmuller, J. J. (2009). Performance measurement in simulation-based training: A review and best practices. *Simulation and Gaming: An Interdisciplinary Journal*, 40(3), 328–376.

Scheirer, J., Klein, J., Fernandez, R., and Picard, R.W. (2002). Frustrating the User on Purpose: A Step Toward Building an Affective Computer. *Interaction with Computers*. *14*(2), 93-118.

Soller, A. (2001). Supporting social interaction in an intelligent collaborative learning system. International Journal of Artificial Intelligence in Education, 12(1), 40-62.

Soller, A., Linton, F., Goodman, B., and Gaimari, R. (1998). Promoting effective peer interaction in an intelligent collaborative learning environment. *In Proceedings of the 4th International Conference on Intelligent Tutoring Systems (ITS '98)*, San Antonio, TX, 186-195.

Sottilare, R., Holden, H., Brawner, K. & Goldberg, B. (2011). Challenges and Emerging Concepts in the Development of Adaptive, Computer-based Tutoring Systems for Team Training. In Proceedings of the Interservice/Industry Training Systems & Education Conference, Orlando, Florida, December 2011.

Sottilare, R.A., Brawner, K.W., Goldberg, B.S. and Holden, H.K. (2012). The Generalized Intelligent Framework for Tutoring (GIFT). Orlando, FL: U.S. Army Research Laboratory – Human Research & Engineering Directorate (ARL-HRED).

Sottilare, R. (2012). Considerations in the development of an ontology for a Generalized Intelligent Framework for Tutoring. *International Defense & Homeland Security Simulation Workshop* in Proceedings of the I3M Conference. Vienna, Austria, September 2012.

Stout, R. J., Salas, E. & Carson, R. (1994). Individual task proficiency and team process: What's important for team functioning. *Military Psychology, 6*, 177-192.

Stout, R.J. Cannon-Bowers, J., Salas, E., Milanovich, D. (1999). Planning, shared mental models, and coordinated performance: An empirical link is established. *Human Factors*, *41*, 61-71.

Suppes, P. & Morningstar, M. (1972). Computer-assisted instruction at Stanford 1966-68: Data, models, and evaluation of the arithmetic programs. New York: Academic Press.

Thompson, J.D. (1967). *Organizations in action*. New York: McGraw-Hill.

Tchounikine, P., Rummel, N. & McLaren, B.M. (2010). "Computer Supported Collaborative Learning and Intelligent Tutoring Systems." In R. Nkambo, J.Bourdeau & R. Mizoguchi (Eds.) *Advances in Intelligent Tutoring Systems*. Chapter 22, 447-463, Springer.

Uttal, W. R. (1962). On Conversational Interaction. In, J. E. Coulson (Ed.) *Programmed Learning and Computer-Based Instruction* (pp. 171-190). New York, NY: John Wiley and Sons.

Van Campen, J. (1981). A computer-assisted course in Russian. In, P. Suppes (Ed.), *University-Level Computer-Assisted Instruction at Stanford: 1968-1980.* Stanford, CA: Institute for Mathematical Studies in the Social Sciences, Stanford University.

Van de Ven, A.H., Delbecq, A.L., and Koenig, R. (1976). Determinants of coordination modes within organizations. *American Sociological Review*, 41, 322-338.

VanLehn, K. (2011). "The Relative Effectiveness of Human Tutoring, Intelligent Tutoring Systems, and Other Tutoring Systems." *Educational Psychologist 46(4)*: 197–221.

VanLehn, K., Lynch, C., Schulze, K., Shapiro, J. A., Shelby, R., Taylor, L. & Wintersgill, M. 2005. "The Andes Physics Tutoring System: Lessons Learned." *International Journal of Artificial Intelligence in Education* 15(3): 147–204.

Vygotsky, L.S. (1978). Mind in Society: The development of higher psychological processes. Cambridge, MA: Harvard University Press.

Warkentin, M., Sayeed, L., and Hightower, R. (1997). Virtual teams versus face-to-face teams: An exploratory study of a web-based conference system. *Decision Sciences*, 28(4), 975-996.

Wegner, D.M. (1986). Transactive memory: A contemporary analysis of the group mind. In G. Mullen & G. Goethals (Eds.), *Theories of Group Behavior* (pp. 185-208). New York: Springer-Verlag.

Yerkes, R.M. & Dodson, J.D. (1908) The relation of strength of stimulus to rapidity of habit-formation. *Journal of Comparative Neurology and Psychology*, *18*, 459-482.

BIOGRAPHIES

Editors

Arthur C. Graesser

Dr. Graesser is a professor in the Department of Psychology and the Institute of Intelligent Systems at the University of Memphis and is an Honorary Research Fellow at the University of Oxford. His primary research interests are in cognitive science, discourse processing, and the learning sciences. More specific interests include knowledge representation, question asking and answering, tutoring, text comprehension, inference generation, conversation, reading, memory, emotions, computational linguistics, artificial intelligence, human-computer interaction, learning technologies with animated conversational agents (such as AutoTutor and Operation ARA), and automated analyses of texts at multiple levels (such as Coh-Metrix, and Question Understanding AID [QUAID]). He served as editor of the journal *Discourse Processes* (1996–2005) and *Journal of Educational Psychology* (2009–2014). His service in professional societies includes president of the Empirical Studies of Literature, Art, and Media (1989–1992), the Society for Text and Discourse (2007-2010), the International Society for Artificial Intelligence in Education (2007–2009), and the Federation of Associations for Behavioral and Brain Sciences Foundation (2012–2013). In addition to receiving major lifetime research achievements awards from the Society for Text and Discourse and University of Memphis, he received an award in 2011 from American Psychological Association on Distinguished Contributions of Applications of Psychology to Education and Training.

Heather Holden

Dr. Holden is currently a researcher in the Learning in Intelligent Tutoring Environments (LITE) Lab within Human Research and Engineering Directorate (HRED) at the U.S. Army Research Laboratory (ARL) in the Simulation and Training Technology Center (STTC) in Orlando, Florida. The focus of her research is in learner modeling, artificial intelligence, and computer-based tutoring system application to education and training. Her research interests also include technology acceptance and human-computer interaction. Dr. Holden previously served as an Information Technology Specialist for the Social Security Administration (SSA) National Computing Center in Woodlawn, Maryland. Dr. Holden earned her Doctorate and Masters in Information Systems from the University of Maryland, Baltimore County. She also has a graduate certificate in Instructional Technology from the same university. Her doctoral research evaluated the relationship between teachers' technology acceptance and usage behaviors to better understand the perceived usability and use of job-related technologies. Her work has been published in the *Journal of Research on Technology in Education*, the *International Journal of Mobile Learning and Organization*, the *Interactive Technology and Smart Education Journal*, and several relevant conference proceedings. Her doctoral work has been continued by other researchers in academia. Dr. Holden also possesses a BS in computer science from the University of Maryland, Eastern Shore.

Xiangen Hu

Dr. Xiangen Hu is a professor in the Department of Psychology at The University of Memphis (UoM) and senior researcher at the Institute for Intelligent Systems (IIS) at the UofM and visiting professor at Central China Normal University (CCNU). Dr. Hu received his MS in applied mathematics (1985) from Huazhong University of Science and Technology, MA in social sciences (1991) and Ph.D. in cognitive sciences (1993) from the University of California, Irvine. Currently, Dr. Hu is the director of the cognitive psychology at the UofM, the Director of Advanced Distributed Learning (ADL) Center for Intelligent Tutoring Systems (ITS) Research & Development, and senior researcher in the Chinese Ministry of Education's Key Laboratory of Adolescent Cyberpsychology and Behavior. Dr. Hu's primary research areas include mathematical psychology, research design and statistics, and cognitive psychology. More specific research interests include general processing tree (GPT) models, categorical data analysis, knowledge representation, computerized tutoring, and advanced distributed learning. Dr. Hu receives funding for the above research from the U.S. National Science Foundation (NSF), U.S. Institute for Education Sciences (IES), ADL of the U.S. Department of Defense (DoD), U.S. Army Medical Research Acquisition Activity (USAMRAA), U.S. Army Research Laboratories (ARL), U.S. Office of Naval Research (ONR), UofM, and CCNU.

Robert Sottilare

Dr. Robert Sottilare serves as the Chief Technology Officer (CTO) of the Simulation & Training Technology Center (STTC) within the Human Research and Engineering Directorate (HRED) at the U.S. Army Research Laboratory (ARL). He also leads adaptive tutoring research within ARL's Learning in Intelligent Tutoring Environments (LITE) Laboratory where the focus of his research is in automated authoring, instructional management, and analysis tools and methods for intelligent tutoring systems. His work is widely published and includes recent articles in the *Cognitive Technology* and the *Educational Technology Journals*. Dr. Sottilare is a co-creator of GIFT (www.GIFTtutoring.org). He received his doctorate in modeling and simulation from the University of Central Florida with a focus in intelligent systems. In January 2012, he was honored as the inaugural recipient of the U.S. Army Research Development & Engineering Command's Modeling & Simulation Lifetime Achievement Award.

Authors

Vincent Aleven

Dr. Vincent Aleven is an Associate Professor in Carnegie Mellon's Human-Computer Interaction Institute, and has 20 years of experience in research and development of advanced learning technologies based on cognitive science theory. His research focuses on metacognition, authoring tools, and the use of tutoring technology in ill-defined domains. Dr. Aleven and colleagues created CTAT, a suite of efficient, easy-to-learn, and easy-to-use authoring tools for intelligent tutoring systems, including a new paradigm called "example-tracing tutors." CTAT is living proof that end-user programming techniques can dramatically increase the cost- effectiveness of tutor authoring even for non-programmers. Dr. Aleven is a member of the Executive Committee of the Pittsburgh Science of Learning Center (PSLC), an national Science Foundation (NSF)-sponsored research center spanning Carnegie Mellon and the University of Pittsburgh. He is a co-founder of Carnegie Learning, Inc., a Pittsburgh-based company that markets Cognitive Tutor™ math courses. He was the Program Committee Co-Chair of the 2010 International Conference on Intelligent Tutoring Systems. He is co-editor in chief of the *International Journal of Artificial Intelligence in Education.* He has been or is principal investigator (PI) on seven major research grants and co-PI on eight others. He has authored over 140 peer-reviewed publications.

William Baggett

Dr. Baggett earned a Ph.D. in cognitive psychology from The University of Memphis in 1998. He also holds an MS in computer science and an MBA in management information systems. William is currently a Project Coordinator in the Computer Science Department at The University of Memphis, where he works on DeepTutor. DeepTutor is an intelligent tutoring system, implemented as a web application, which uses natural language dialog to teach conceptual physics to high school and college students. DeepTutor is funded by the Institute of Education Sciences. Previously, William was a professor and part-time department chair of computer information systems at Strayer University and an adjunct professor of computer science at The University of Memphis. In both positions, William taught graduate and undergraduate computer science courses, mentored, tutored, and advised students, and developed new curricula. He was also a business analyst at FedEx Express, where he wrote software specifications for PowerPad, a mission-critical handheld computer carried by FedEx Express couriers. PowerPad software is designed to promote optimal courier behavior including the efficient pickup and delivery of FedEx shipments, package tracking, and conformance to policies and procedures for a wide variety of domestic and international services.

Ryan Shaun Joazeiro de Baker

Dr. Baker is the Julius and Rosa Sachs Distinguished Lecturer at Teachers College, Columbia University. He earned his Ph.D. in human-computer interaction from Carnegie Mellon University. Baker was previously Assistant Professor of Psychology and the Learning Sciences at Worcester Polytechnic Institute, and he served as the first technical director of the Pittsburgh Science of Learning Center DataShop, the largest public repository for data on the interaction between learners and educational software. He is currently serving as the founding president of the International Educational Data Mining Society, and as associate editor of the *Journal of Educational Data Mining*. His research combines educational data mining and quantitative field observation methods in order to better understand how students respond to educational software, and how these responses impact their learning. He studies these issues within intelligent tutors, simulations, multi-user virtual environments, and educational games, within populations from pre-schoolers, to middle school students, to military trainees.

Avron Barr

Mr. Barr is the director of The LETSI Foundation. He’s done research on intelligent tutoring systems at Stanford University, written a four-volume reference book about Artificial Intelligence, and co-founded a Silicon Valley software startup that went public in 1986. Since then, he has advised dozens of companies, startups, government agencies, and Non-Government Organizations (NGOs) about bringing innovative software products to market. During the 1990s, he co-directed a study of the global software industry on a Sloan Foundation grant at Stanford’s Graduate School of Business. In recent years, he’s advised the Defense Advanced Research Projects Agency (DARPA) DARWARS project on serious games; served as Strategic Director for SCORM at the Department of Defense’s (DoD) Advanced Distributed Learning Initiative; and helped start LETSI’s work on open software standards to support an era of rapid evolution in educational technology. He teaches a freshman seminar on *The Business of the Internet* at Stanford, and spends his spare time hiking in the redwood forests around Santa Cruz, California.

Keith Brawner

Dr. Brawner is a researcher for the Learning in Intelligent Tutoring Environments (LITE) Lab within the Human Research & Engineering Directorate (HRED) at the U. S. Army Research Laboratory (ARL). He has 7 years of experience within U.S. Army and Navy acquisition, development, and research agencies. He holds a doctoral degree in computer engineering with a focus on intelligent systems and machine learning from the University of Central Florida. The focus of his current research is in machine learning, active learning, real-time processing, datastream mining, adaptive training, affective computing, and semi/fully automated user tools for adaptive training content.

Winslow Burleson

Dr. Burleson is an assistant professor of human computer interaction in the School of Computing, Informatics, and Decision Systems Engineering at Arizona State University (ASU) and a Visiting Honors Faculty Fellow in ASU's Barrett Honors College. He is the Founding Director of the Motivational Environments research group and author of over 100 scientific publications (including the "best paper" at AI in Ed 2009 and the 2011 UMUAI James Chen Award) and has been awarded 10 patents. In 2009, he was recognized by the National Academy of Engineering as, "One of the nation's brightest young engineering researchers and educators." He received his Ph.D. from the Massachusetts Institute of Technology (MIT) Media Lab, is a Kavli Fellow, and serves on National Academy of Engineering, National Academies of Sciences, and National Science Foundation committees. Burleson has a B.A. in biophysics from Rice University and an MSE in mechanical engineering from Stanford University.

Whitney Cade

Ms. Cade is an experimental psychology doctoral student at the University of Memphis with a concentration in cognitive psychology. She graduated from Rhodes College with a B.A. in psychology. Working primarily in the Institute for Intelligent Systems, her primary interests include cognitive science and learning. Her research specifically focuses on expert human tutoring, large-grain pedagogical strategies, intelligent tutoring systems, pedagogical agents, image display techniques that support learning, and the application of machine learning in educational software.

Zhiqiang Cai

Mr. Cai is a research assistant professor with the Institute for Intelligent Systems at the University of Memphis. He has a masters of science degree in computational mathematics received in 1985 from Huazhong University of Science and Technology, P. R. China. His current research interests are in algorithm design and software development for tutoring systems and natural language processing.

Cristina Conati

Dr. Conati is an associate professor of computer science at the University of British Columbia, Vancouver, Canada. She received a "Laurea" degree (M.Sc. equivalent) in computer science at the University of Milan, Italy (1988), as well as a M.Sc. (1996) and Ph.D. (1999) in intelligent systems at the University of Pittsburgh. Dr. Conati's research goal is to integrate research in artificial intelligence, cognitive science, and human-computer interaction to make complex interactive systems increasingly more effective and adaptive to the users' needs. Her areas of interest include intelligent user interfaces, user modeling, user-adaptive systems, and affective computing. Her research has received awards from the International Conference on User Modeling, the International Conference of AI in Education, the International Conference on Intelligent User Interfaces (2007), and the *Journal of User Modeling and User Adapted Interaction* (2002). Dr. Conati is an associate editor for the *Journal of AI in Education*, for the *IEEE Transactions on Affective Computing*, and the *ACM Transactions on Intelligent Interactive Systems*.

Mark Conley

Dr. Conley is a professor of literacy at the University of Memphis. He specializes in human and technology-based tutoring and assessment in literacy, mathematics, and science. He designed the Memphis Literacy Corps, the largest literacy tutoring effort implemented in the United States. Currently, he is designing tutoring programs for adult learners in community-based tutoring programs, including a program for potentially incarcerated youth. He has published numerous articles about literacy, disciplinary literacy, teacher education, and assessment in journals like the *Harvard Educational Review*, *Theory Into Practice*, and the *Journal of Adolescent and Adult Literacy*. He has published several books about disciplinary literacy, curriculum standards, and assessment. He has also advised officials at state, national, and international levels on curriculum standards and assessment. He holds a Certified Flight Instructor certificate with an instrument rating and teaches and practices guitar building as part of his passion for teaching and learning.

Scott Douglass

Dr. Douglass is a senior research psychologist with the 711/HPW Cognitive Models and Agents Branch (RHAC), U.S. Air Force Research Laboratory, Wright-Patterson Air Force Base, Ohio. He holds a Ph.D. (2007) in cognitive psychology from Carnegie Mellon University. Working with John R. Anderson at CMU, he acquired expertise in cognitive architectures, eye-tracking systems, and the modeling and simulation of complex situated cognitive processes. His research interests include large-scale cognitive modeling in event-driven computer-based systems, complex event processing, artificial intelligence, knowledge representation, multi-formalism modeling, and automated training aids. He is a member of the Society for Modeling and Simulation International (SCS). He can be reached at scott.douglass@wpafb.af.mil.

Paula Durlach

Dr. Durlach received her Ph.D. in experimental psychology from Yale University in 1982, and subsequently held fellowship positions at the University of Pennsylvania and the University of Cambridge. From 1987 to 1994, she was an assistant professor of psychology at McMaster University and then went on to lead the exploratory consumer science team at Unilever Research Colworth Laboratory in the U. K. She returned to the U. S. in 2001 to join the U. S. Army Research Institute for the Behavioral and Social Sciences. Since April 2012, she has been the Deputy Director of the Advanced Distributed Learning Initiative. Dr. Durlach has received recognition for her work in experimental psychology and cognitive science at the Army Science Conference and from the Department of Army Research and Development. She is a Fellow of the Association for Psychological Science, and member of the Experimental Psychology Society, the Psychonomic Society, and the Society for Artificial Intelligence in Education. With Dr. Alan Lesgold, she co-edited the book, *Adaptive Technologies for Training and Education*, published in 2012, and has also published research in journals such as *International Journal of Artificial Intelligence in Education*, *Military Psychology*, *Computers in Human Behavior*, and *Human-Computer Interaction*.

Dexter Fletcher

Dr. Fletcher is a member of the senior research staff at the Institute for Defense Analyses where he specializes in personnel and human performance issues. His graduate degrees are in computer science and educational psychology from Stanford University, where, as a research associate, he directed projects for the Institute for Mathematical Studies in the Social Sciences. He has held university positions in psychology, computer science, and systems engineering and government positions in Navy and Army Service Laboratories, the Defense Advanced Research Projects Agency, and the White House Office of

Science and Technology Policy. He has served on science and technology advisory panels for the Defense Science Board, Army Science Board, Naval Studies Board, Air Force Scientific Advisory Board, National Science Foundation, National Academy of Sciences, and the National Academy of Engineering. He has designed computer-based instruction programs used in public schools and training devices used in military training. He is a Fellow of the American Educational Research Association and three divisions of the American Psychological Association. His research interests include intelligent tutoring systems, synthetic environments in education and training, mobile performance aids, analyses of skilled behavior, and cost-effectiveness analyses of education and training.

Donald Franceschetti

Dr. Franceschetti received his B.S. degree in 1969 in chemistry from Brooklyn College and the Ph.D. degree in physical chemistry from Princeton University in 1972. Following two postdoctoral appointments in physics departments (Urbana, IL and Chapel Hill, NC), he joined the department of physics at the then Memphis State University in 1979. Currently he is Dunavant University Professor of Physics and Chemistry at the University of Memphis. His research interests have ranged from the properties of liquid mixtures to spin-lattice relaxation in solids, ionic mobility in solids and x-ray states in metals and atoms. He has also worked on molecular electronic structure of biological molecules. Dr. Franceschetti has published about 70 research papers and has done several dozen presentations at scientific meeting. He has authored more than 100 articles on science and mathematics appearing in reference works for young people. He has been involved in developing intelligent tutoring systems for conceptual physics since the early 1990s. He has also been PI or co-PI of grants totaling about $10 million.

Elizabeth Gire

Dr. Gire is an assistant professor of physics at the University of Memphis and conducts research in the area of physics education. Her research interests are in the areas of sense-making and problem-solving skills and representational fluency. As a graduate student working with Professor Barbara Jones at the University of California, San Diego, she investigated the epistemological development and the development of problem solving strategies of undergraduate physics majors. This work looked at students across the undergraduate curriculum, from freshman to senior level students. While working with Professor Corinne Manogue at Oregon State University, Dr. Gire studied students' understandings of quantum operators and student reasoning about electrostatic phenomena. She has had intensive training in teaching and developing curricular materials in the Paradigms curriculum, which uses interactive teaching strategies and a unique content ordering and structure to help students think about physics content the way that professional physicists do. At Kansas State University, she worked with Dr. Sanjay Rebello to study introductory physics students' representational fluency, and tracked the development of this fluency over the course of a two semester introductory sequence. Additionally, she worked on several other projects, including research on student reasoning about simple machines when interacting with real or virtual experimental apparatus, the correlation between visual attention and conceptual reasoning in physics, and students' strategies for solving ill-structured problems in an advanced undergraduate electronics course. She has published several articles in peer-reviewed journals and many peer-reviewed conference proceedings.

Benjamin Goldberg

Dr. Goldberg is a member of the Learning in Intelligent Tutoring Environments (LITE) Lab at Human Research & Engineering Directorate (HRED) at the U.S. Army Research Laboratory's (ARL) Simulation and Training Technology Center (STTC) in Orlando, Florida. He has been conducting research in the modeling and simulation community for the past five years with a focus on adaptive learning and how to leverage artificial intelligence tools and methods for adaptive computer-based instruction. Currently, he is the LITE Lab's lead scientist on instructional strategy research within adaptive training environments. He explores the development and integration of tools and methods for delivering tailored training experiences, and identifies strategies of interest for empirical evaluation to assess their effectiveness across multiple domains. Dr. Goldberg holds doctoral and masters degrees in modeling & simulation from the University of Central Florida. Prior to employment with ARL, he held a graduate research assistant position for two years in the Applied Cognition and Training in Immersive Virtual Environments (ACTIVE) Lab at the Institute for Simulation and Training. Dr. Goldberg's work has been published across several well-known conferences, with recent contributions to both the Human Factors and Ergonomics Society (HFES) and Intelligent Tutoring Systems (ITS) proceedings.

Ken Koedinger

Dr. Koedinger is professor of human-computer interaction and psychology at Carnegie Mellon University (CMU). His research has contributed new principles and techniques for the design of educational software and has produced basic cognitive science research results on the nature of mathematical thinking and learning. Dr. Koedinger is a co-founder of Carnegie Learning (carnegielearning.com) and the CMU Director of LearnLab (learnlab.org). LearnLab is supporting Big Data investigations in education and, more generally, leverages cognitive and computational approaches to support researchers in investigating the instructional conditions that cause robust student learning.

Alan Lesgold

Dr. Lesgold is professor and, since July 2000, dean of the School of Education at the University of Pittsburgh and also professor of psychology and intelligent systems. He received his Ph.D. in psychology from Stanford University in 1971 and also holds an honorary doctorate from the Open University of the Netherlands. He is a fellow of the American Psychological Association (APA), in experimental, applied, and educational psychology, and also of the Association for Psychological Science and the American Educational Research Association. In 2001, he received the APA award for distinguished contributions of applications of psychology to education and training. In 1995, he was awarded the Educom Medal. He was president of the Applied Cognitive Psychology division of the International Association for Applied Psychology 2002-2006. Lesgold is a Lifetime National Associate of the National Research Council (National Academies). He also was appointed by Governor Rendell as a member of the Governor's Commission on Preparing America's Teachers in 2005 and served later on the State's commission on cyber high schools as well. He served as chair of the National Research Council committee on adolescent and adult literacy. In that role, he led an extensive study of the systems, both in the K–12 world and in community colleges, for remediating literacy problems. From 1986 to 2000, he was executive associate director of the Learning Research and Development Center at the University of Pittsburgh. He is on the board of Teaching Matters. Lesgold has served the Pittsburgh community mostly in education-related activities, including board service for A+ Schools, Youthworks (completed), and the Pittsburgh Regional Center for Science Teaching. He serves as chair of the advisory board for the Center for Learning at Community College of Allegheny County. He was president of Rodef Shalom Congregation in 2002–2004. He is married to Sharon Lesgold, a retired mathematics educator, and they have two grown sons, Jacob and Noah.

James Lester

Dr. Lester is Distinguished Professor of Computer Science at North Carolina State University. His research focuses on transforming education with technology-rich learning environments. Using artificial intelligence, game technologies, and computational linguistics, he designs, develops, fields, and evaluates next-generation learning technologies for K–12 science, literacy, and computer science education. His work on personalized learning ranges from game-based learning environments and intelligent tutoring systems to affective computing, computational models of narrative, and natural language tutorial dialogue. He received his B.A. (Highest Honors, Phi Beta Kappa), M.S.C.S., and Ph.D. in computer science from the University of Texas at Austin. He received his B.A. in history from Baylor University. He has served as Program Chair for the International Conference on Intelligent Tutoring Systems, the International Conference on Intelligent User Interfaces, and the International Conference on Foundations of Digital Games, on the editorial board of *Metacognition and Learning*, and as editor-in-chief of the *International Journal of Artificial Intelligence in Education*. He has been recognized with a National Science Foundation CAREER Award and several Best Paper Awards.

Antonija (Tanja) Mitrovic

Dr. Mitrovic is a full professor and the Head of the Department of Computer Science and Software Engineering at the University of Canterbury, Christchurch, New Zealand. She is the leader of the Intelligent Computer Tutoring Group (ICTG). Dr Mitrovic received her Ph.D. in computer science from the University of Nis, Yugoslavia, in 1994. Prof Mitrovic is president-elect of the International Society of Artificial Intelligence in Education. She is an associate editor of the following journals: *International Journal on Artificial Intelligence in Education*, I*EEE Transactions on Teaching and Learning Technologies*, and *Research and Practice in Technology Enhanced Learning* (RPTEL). Dr. Mitrovic's primary research interests are in student modeling. ICTG has developed a number of constraint-based intelligent tutoring systems in a variety of domains, which have been thoroughly evaluated in real classrooms, and proven to be highly effective. These systems provide adaptive support for acquiring both problem-solving skills and meta-cognitive skills (such as self-explanation and self-assessment). Although most of the ITSs developed by ICTG support students learning individually in areas such as database querying (SQL-Tutor), database design (EER-Tutor and ERM-Tutor), data normalization (NORMIT), there are also constraint-based tutors for object-oriented software design and collaborative skills, various engineering topics (thermodynamics, mechanics), training to interpret medical images and language-learning. ICTG has also developed ASPIRE, a full authoring and deployment environment for constraint-based tutors. Recent research includes affect-aware tutors and motivational tutors.

Donald "Chip" Morrison

Dr. Morrison is a faculty affiliate at IIS. A graduate of Dartmouth, Dr. Morrison holds an M.A. from the University of Hong Kong and an Ed.D. from Harvard. Dr. Morrison has more than 40 years experience in education, including stints teaching English as a Second Language, as an educational software developer, and museum exhibit developer. As a senior scientist at Bolt, Beranek and Newman, Dr. Morrison helped found Co-nect, a comprehensive school reform model. Among other contributions, he established and directed the Co-nect Critical Friends Program, a review process used by hundreds of schools nationwide. After leaving Co-nect, he joined the Tripod Project, part of the Achievement Gap Initiative at Harvard's Kennedy School of Government, and spent a year as lead School Quality Reviewer in New York City. Since coming to the University of Memphis in 2008, he has helped bring in grants and contracts worth more than $500K annually, including a five-year, $3.5M grant from the U.S. Department of Education to evaluate a large-scale science education initiative run by the Smithsonian Institution. His current research interests include models of human cognition and learning, and the application of these models in the design of conversation-based intelligent learning systems.

Bradford Mott

Dr. Mott is a senior research scientist in the Department of Computer Science at North Carolina State University. Prior to joining North Carolina State University, he served as Technical Director at Emergent Game Technologies where he created cross-platform middleware solutions for video game development, including solutions for the PlayStation 3, Wii, and Xbox 360. Dr. Mott received his Ph.D. in computer science from North Carolina State University in 2006, where his research focused on intelligent game-based learning environments. His current research interests include computer games, computational models of interactive narrative, and intelligent game-based learning environments.

Kasia Muldner

Dr. Muldner received her Ph.D. from the Department of Computer Science at the University of British Columbia, where she designed and evaluated a computational tutor that supported students during analogical problem solving. She is currently a post-doctoral researcher in the Department of Computing, Informatics, and Decision Systems Engineering at Arizona State University. Her work falls into the intersection of human-computer-interaction and artificial intelligence, dealing with the design and evaluation of interactive educational technologies that aim to help students learn effectively though personalized support. She is particularly interested in technologies that support high level student states related to meta-cognition, affect, and creativity.

Rodney Nielsen

Dr. Nielsen received a dual Ph.D. in computer science and cognitive science from the University of Colorado, Boulder in 2008. He is currently an associate professor of computer science and engineering at UNT, where he co-directs the Language and Information Technologies (LIT) lab. Prior to UNT, he was an assistant professor adjunct of computer science at CU Boulder, a research scientist in CU's Center for Computational Language and Education Research, and a research scientist with Boulder Language Technologies. Dr. Nielsen's research includes the areas of natural language processing, machine learning, and cognitive science, with an emphasis on educational technology (classroom engagement technology and intelligent tutoring systems), spoken-dialogue educational health and wellbeing companion robots (companionbots), health and clinical informatics, and end-user software engineering. One common theme across all of these areas is the need for a robust, informative model of the humans interacting with the system. Further information regarding Dr. Nielsen's research and contact information can be found at http://www.cse.unt.edu/~nielsen/.

Andrew Olney

Dr. Olney is presently an assistant professor in the Institute for Intelligent Systems/Department of Psychology at the University of Memphis and Associate Director of the Institute for Intelligent Systems. Dr. Olney received a B.A. in linguistics with cognitive science from University College London in 1998, an M.S. in evolutionary and adaptive systems from the University of Sussex in 2001, and a Ph.D. in computer science from the University of Memphis in 2006. Dr. Olney's primary research interests are in natural language interfaces. Specific interests include vector space models, dialogue systems, unsupervised grammar induction, robotics, and intelligent tutoring systems. Dr. Olney is co-chair of the International Educational Data Mining Society Annual Conference (EDM2013) and frequently serves as program committee member and journal reviewer in the fields of cognitive science, artificial intelligence, and education. Together with his collaborators, Dr. Olney has been awarded $8.6 million from federal funding agencies including the National Science Foundation, the Institute for Education Sciences, and the Department of Defense. His research has been featured in *WIRED Magazine*, the *New York Times*, the *Wall Street Journal*, the Discovery Science Channel, and BBC Radio 4. Dr. Olney was awarded first place in an international robotics competition for the PKD Android (AAAI, 2006) and received the Early Career Research Award from the University of Memphis.

Philip I. Pavlik Jr.

Dr. Pavlik is currently an assistant professor of psychology at the University of Memphis Institute for Intelligent Systems. Dr. Pavlik received a B.A. from the University of Michigan in economics and a Ph.D. from Carnegie Mellon University where he studied cognitive psychology with John Anderson (developer of the ACT-R cognitive modeling system) and received a neuroscience certificate from the Center for the Neural Basis of Cognition. With Anderson, Dr. Pavlik has pioneered changes in the ACT-R theory that have allow his research to use this theory to quantitatively optimize the learning of information for tasks such as flashcard learning. From this foundation, his work with Dr. Koedinger has developed to focus on problem solving, schema learning, optimal transfer, effects of motivational constructs, and student strategy use. His methodologies include theory development, experimentation, mathematical modeling, and educational applications. Dr. Pavlik has received more than 1.5 million dollars in grant awards from the Institute for Educational Sciences, the National Science Foundation, and other sources.

Elaine Raybourn

Dr. Raybourn has a Ph.D. in intercultural communication with an emphasis in human-computer interaction. She is a principal member of the technical staff in cognitive systems at Sandia National Laboratories.* Elaine led the development of an award-winning Government game identified by a 2006 Defense Science Board Study on 21st Century Strategic Technology Vectors as “critical capabilities and enabling technologies for the 21st century that show promise.” She was an European Consortium for Research in Informatics and Mathematics (ERCIM) Fellow and a recipient of the Department of the Army Award for Patriotic Civilian Service, awarded to her by the U.S. Army Special Forces. Dr. Raybourn serves on advisory and editorial boards of the international journals *Interactive Technology and Smart Education*, *Journal of Game-based Learning*, and *Simulation & Gaming*. She is on assignment to Advanced Distributed Learning Initiative, Office of the Deputy Secretary of Defense (Readiness), where she leads research on transmedia learning, distributed cognition, learner adaptability, and next generation learners’ interactions with personalized assistants for learning (PAL).

*Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy’s National Nuclear Security Administration under contract DE-AC04-94AL85000.

Damon Regan

Dr. Regan is a contractor with The Tolliver Group, Inc. and supports the Advanced Distributed Learning (ADL) Initiative as a technical team co-lead. He has contributed to efforts of the ADL Initiative since 2004 including most recently the Personal Assistant for Learning (PAL) and the Training and Learning Architecture (TLA). Damon completed his Ph.D. in instructional technology from the University of Central Florida. He also has an MBA from Rollins College and a B.S. in computer science from the University of Central Florida.

Robby Robson

Dr. Robson is a learning technology researcher, innovator, and entrepreneur. His first career was in mathematics in the areas of real algebraic geometry and computational number theory. In 1995, he began developing web-based learning environments and technologies, including one of the first online calculus courses. Soon thereafter he became involved in developing industry standards for eLearning interoperability, contributing to several of the early IMS and IEEE standards and chairing the IEEE Learning Technology Standards Committee from 2000–2008. Dr. Robson co-founded Eduworks in 2001 where he is CEO and chief scientist. Over the past dozen years he guided Eduworks while helping numerous commercial, academic, and non-profit organizations design and develop educational and training technology and formulate market strategies while serving as the principle investigator or lead scientist on multiple National Science Foundation and Department of Defense projects related to repositories, open content formats, competency management, and intelligent tutoring systems. His most recent work is in applications of semantic technology and natural language processing. He holds a doctorate from Stanford University in mathematics and has held posts in both academia and industry.

Lisa Rossi

Ms. Rossi works as a consultant through the Department of Social Science and Policy Studies at Worcester Polytechnic Institute (WPI). She currently studies at Georgia Institute of Technology working toward a master's degree in human-computer interaction. Previously, she received her bachelor's degree in psychological science and humanities & arts (music concentration) from WPI, after which she worked at WPI as a research analyst in the Educational Psychology and Learning Sciences Laboratories assisting with research projects focused primarily around intelligent tutoring systems for scientific inquiry skills used by middle school students.

Jonathan Rowe

Dr. Rowe is a senior research associate in the Department of Computer Science at North Carolina State University. He received his Ph.D. and M.S. degrees in computer science from North Carolina State University. He received his B.S. degree in computer science from Lafayette College. His research is in the areas of artificial intelligence and human-computer interaction for advanced learning technologies, with an emphasis on game-based learning environments. He is particularly interested in intelligent tutoring systems, user modeling, educational data mining, and computational models of interactive narrative. Mr. Rowe has led development efforts on several game-based learning projects, including Crystal Island: Lost Investigation, which was nominated for Best Serious Game at the 2012 Unity Awards and the 2012 I/ITSEC Serious Games Showcase and Challenge. His research has also been recognized with several best paper awards, including best paper at the Seventh International Artificial Intelligence and Interactive Digital Entertainment Conference and best paper at the Second International Conference on Intelligent Technologies for Interactive Entertainment.

Vasile Rus

Dr. Rus is an associate professor of computer science with a joint appointment in the Institute for Intelligent Systems (ITS) whose areas of expertise are computational linguistics, artificial intelligence, software engineering, and computer science in general. His research areas of interest include question answering and asking, dialogue-based intelligent tutoring systems (ITSs), knowledge representation and reasoning, information retrieval, and machine learning. For the past 10 years, Dr. Rus has been heavily involved in various dialogue-based ITS projects including systems that tutor students on science topics (DeepTutor), reading strategies (iSTART), writing strategies (W-Pal), and metacognitive skills (MetaTutor). Currently, Dr. Rus leads the development of the first intelligent tutoring system based on learning progressions, DeepTutor (www.deeptutor.org). He has coedited two books, received several Best Paper Awards, and authored more than 90 publications in top, peer-reviewed international conferences and journals. He is currently Associate Editor of the *International Journal on Artificial Intelligence Tools*.

Jennifer Sabourin

Ms. Sabourin is currently a Ph.D. student at North Carolina State University in the Department of Computer Science. She received her B.S. (2008) and M.S. (2012) in computer science from North

Carolina State University where she graduated as Valedictorian. She is a recipient of the National Science Foudation Graduate Research Fellowship award. Her research interests include artificial intelligence and its application in education. In particular, her research focuses on modeling student affect, engagement, self-regulation, and problem solving in game-based learning environments. Her work has been recognized with a best student paper award at the third International Conference on Affective Computing and Intelligent Interaction.

Valerie Shute

Dr. Shute is the Mack & Effie Campbell Tyner Endowed Professor in Education in the Department of Educational Psychology and Learning Systems at Florida State University. Before coming to FSU in 2007, she was a principal research scientist at Educational Testing Service where she was involved with basic and applied research projects related to assessment, cognitive diagnosis, and learning from advanced instructional systems. Her general research interests hover around the design, development, and evaluation of advanced systems to support learning – particularly related to 21st century competencies. An example of current research involves using immersive games with stealth assessment to support learning – of cognitive and non-cognitive knowledge, skills, and dispositions. Her research has resulted in numerous grants, journal articles, books, chapters in edited books, a patent, and a 2010 book co-edited with Betsy Becker entitled, *Innovative assessment for the 21st century: Supporting educational needs.*

Anne Sinatra

Dr. Sinatra is an Oak Ridge Associated Universities Post Doctoral Fellow in the Learning in Intelligent Tutoring Environments (LITE) Lab at the U.S. Army Research Laboratory's (ARL) Simulation and Training Technology Center (STTC) in Orlando, FL. The focus of her research is in cognitive and human factors psychology. She has specific interest in how information relating to the self and about those that one is familiar with can aid in memory, recall, and tutoring. Her dissertation research evaluated the impact of using degraded speech and a familiar story on attention/recall in a dichotic listening task. Her work has been published in the journal *Interaction Studies*, and in the conference proceedings of the Human Factors and Ergonomics Society. Prior to becoming a post doc, Dr. Sinatra was a graduate research associate with UCF's Applied Cognition and Technology (ACAT) Lab, and taught a variety of undergraduate psychology courses. Dr. Sinatra received her Ph.D. and M.A. in applied experimental and human factors psychology, as well as her B.S. in psychology from the University of Central Florida.

Matthew Small

Dr. Matthew Small is a research consultant and educational game developer working with various educational research entities including the College of Education at The Florida State University. His research interests center on re-designing the template for modern educational game development through embedding mechanisms for learning and assessment into engaging video games that rely on nonlinear gameplay mechanics. His work focuses on STEM areas of content knowledge that are traditionally considered difficult to teach and measure in "fun" digital environments. Dr. Small has collaboratively developed educational software including *Newton's Playground* and *Codecraft*, games that teach quantitative physics and computational thinking, respectively, and are the focus of numerous publications as well as ongoing research.

Abhiraj Tomar

Mr. Tomar received his B.S. in computer science from BITS Pilani University, India and is currently working as a research intern at the University of North Texas. He completed a B.S. thesis on user modeling for companionbots. His work aims at developing dialog based healthcare agents to provide real-time care to elderly and depressed. For an undergraduate project, he worked on search engine optimization and developed a clustering algorithm using latent semantic analysis and graph theory. He has also worked on optimization of downstream processing and online question bank organizer. His research interests include artificial intelligence, algorithms, graph theory, natural language processing and multidisciplinary applications of computer science. He headed the marketing team of his undergraduate institution for a year and was also a manager for the public relations team.

Matthew Ventura

Mr. Ventura is a senior research scientist in the College of Education at Florida State University. His current research spans both cross-sectional and experimental research around how video games can affect cognitive and noncognitive skills. Specific skills of interest include large-scale spatial ability, problem solving, persistence, conceptual physics, and computational thinking. Mr. Ventura also designs educational video games and simulations that use embedded assessments to enhance learning. His work has resulted in numerous publications spanning the areas of assessment, cognitive science, and education.

Learner Modeling Advisory Board Members

The editors of *Design Recommendations for Intelligent Tutoring Systems - Learner Modeling (Volume 1)* would like to recognize the contributions of learner modeling advisory board members whose contributions to this effort were not in the authoring of book chapters, but provided motivations and focus to the development of this volume.

Trey Martindale

Dr. Martindale is associate professor for the Instructional Design and Technology graduate program with the University of Memphis, and is a research scientist with the Institute for Intelligent Systems. His research expertise is in the design and analysis of online learning environments. His research and development work has been funded by NSF, the Centers for Disease Control, the U.S. Department of Education, the Institute of Education Sciences, IBM, Microsoft, and others. Dr. Martindale has contributed to the creation of four fully online degree programs at three universities, and has developed numerous online courses. He has pioneered the effective pedagogical use of social software tools for enabling collaborative learning environments. He is an active consultant to companies and organizations seeking to improve their employee training and performance. He also leads consulting efforts within the university IDT program, directing IDT graduate students on instructional design and e-learning projects with external clients. He serves on the Board of Directors for the Association for Educational Communications and Technology, and is on the review board for Educational Technology Research and Development, and the Quarterly Review of Distance Education.

Kurt VanLehn

Dr. VanLehn is the Diane and Gary Tooker Chair for Effective Education in Science, Technology, Engineering and Math in the Ira A. Fulton Schools of Engineering at Arizona State University. He received a Ph.D. from MIT in 1983 in computer science, was a post-doc at BBN and Xerox PARC, joined the faculty of CMU in 1985, moved to the University of Pittsburgh in 1990 and joined ASU in 2008. He founded and co-directed two large NSF research centers (Circle; the Pittsburgh Science of Learning Center). He has published over 125 peer-reviewed publications, is a fellow in the Cognitive Science Society, and is on the editorial boards of *Cognition and Instruction* and the *International Journal of Artificial Intelligence in Education*. Dr. VanLehn's research focuses on intelligent tutoring systems and other intelligent interactive instructional technology.

Ray S. Perez

Dr. Perez is a senior scientist and program officer at the Office of Naval Research (ONR) in Arlington, Virginia. In this capacity, he is responsible for and actively manages ONR's Cognitive Science of Learning Program. This Program has three major multidisciplinary and highly intertwined thrusts. Specifically, he is responsible for (1) training/education research and their core technologies, (2) individual differences research and (3) neuro-biology of learning research. Dr. Perez also conducts research in the development of new theories and methods for the assessment of human abilities. He is recognized as a DoD leader in learning, he was asked by the Chief of Naval research to be the program manager for ONR's STEM Grand Challenge. Dr. Perez has conducted research in the areas of technology-based education / training spans over 20 years. Throughout his career, he has received numerous awards for his work in advanced learning technologies for training and education. He has authored four books on the use of technology in education and training. Most recently, he co-edited *Computer Games and Team and Individual Learning Technology* with Dr. Harry O'Neil that is published (2011) by Elsevier. His next book also co-edited with Drs. Harry O'Neil and Eva Baker entitled *Teaching and Measuring Cognitive Readiness* (in preparation) will be published by Springer Publishing Company.

Prior to coming to ONR he served as program manager for the Presidential Technology Initiative Program at the Department of Defense Education Activity (DoDEA). While at DoDEA he was the Director of the K–12 program within the Advanced Distribute Learning Initiative, sponsored by the Office of the Secretary of Defense, Readiness and Training. Earlier, he was principal scientist in Simulation and Advanced Instructional Systems, at the U.S. Army Research Institute for the Social and Behavioral Sciences (ARI) and was an assistant professor, in the Department of Psychology, at California State University Dominguez Hills, California.

Dr. Perez continues to serve as an educational technology expert on various review panels including the National Science Foundation (NSF), National Academy Sciences (NAS), and the Defense Advanced Research Agency (DARPA). Dr. Perez received a Doctorate and Master's degrees in Educational Psychology with emphasis on Cognitive Psychology from the University of California, Los Angeles California.

Beverly Park Woolf

Dr. Woolf is a research professor at the University of Massachusetts who develops intelligent tutors that model student affective and cognitive characteristics and combine cognitive analysis of learning with artificial intelligence, network technology, and multimedia. These systems represent the knowledge taught, recognize learners' skills and behavior, use sensors, and machine learning to model student affect, and adjust problems to help individual students. Dr. Woolf has developed tutors in education and industry and in a variety of disciplines (e.g., chemistry, psychology, physics, geology, art history, mathematics, and economics). Some of these tutors enable students to pass standard exams at a 20% higher rate and one system is used by more than 150,000 students per semester across hundreds of colleges. Dr. Woolf published the book *Building Intelligent Interactive Tutors* along with over 200 articles. She is lead author on the NSF report *Roadmap to Education Technology* in which 40 experts and visionaries identified the next big computing ideas that will define education technology and developed a vision of how technology can incorporate deeper knowledge about human cognition and develop dramatically more effective instructional strategies. Dr. Woolf has delivered keynote addresses, panels, and tutorials in more than 20 foreign countries and is a fellow of the American Association of Artificial Intelligence.

ACRONYM LIST

2-D	two-dimensional
AAWC	anti-air warfare coordinator
ABC	Affective-Behavioral-Cognitve
ACE	Adaptive Coach for Exploration
ACL	agent communication language
ACAT	Applied Cognition and Technology
ACTIVE	Applied Cognition and Training in Immersive Virtual Environments
ADL	Advanced Distributed Learning Initiative
AFRL	Air Force Research Laboratory
AICA	kaike information criterion
ALEKS	Assessment and Learning in Knowledge Spaces
API	application programming interface
ARL	U.S. Army Research Laboratory
ASN	Achievement Standards Network’s
ASU	Arizona State University
BKT	Bayesian knowledge tracing
CBM	constraint-based modeling
CCNU	Central China Normal University
CBTS	computer-based training systems
CDOs	Cognitive doman ontologies
CECEP	complex event processing
CMU	Carnegie Mellon University

CRC	Current Relevant Contribution
CSP	constraint satisfaction problem
CTAT	Cognitive Tutor Authoring Tool
CTO	Chief Technology Officer
DAG	Directed acyclic graph
DARPA	Defense Advanced Research Projects Agency
DKF	Domain Knowledge File
DLs	Domain-specific languages
DOD	U.S. Department of Defense
ECD	Evidence-Centered Design
ECS	Engineering and Computer Simulations’
EDA	electro-dermal activity
EDM	educational data mining
EER	enhanced entity-relationship
EMDAT-RT	Eye Movement Data Analysis Toolkit in Real Time
EMT	Expectation and Misconception Tailored
EP	empirical progression
ERIC	Educational Resource Information Center
EWS	electronic warfare signal
FIPA	Foundation for Intelligent Physical Agents
FYI	for your information
GIFT	Generalized Intelligent Framework for Tutoring
GME	Generic Modeling Environment
GPT	general processing tree
HAI	human-computer interaction
HAL	Hyperspace Analogue to Language

HFES	Human Factors and Ergonomics Society
HRED	Human Research and Engineering Directorate
ICS	Intelligent Creativity Support
IES	U.S. Institute for Education Sciences
IO	input/output
IRT	Item Response Theory
IS	interactive simulation
ISI	Instructional Strategies Indicator
ITA	InternationalTechnology Alliance
ITS	intelligent tutoring system
JADE	Java Agent Development Framework
KCs	knowledge components
KERMIT	Knowledge-based Entity Relationship Modeling Intelligent Tutor
KST	knowledge space theory
LCC	Learner Characteristic Curve
LDA	Latent Dirichlet Allocation
LIT	Language and Information Technologies
LITE	Learning in Intelligent Tutoring Environments
LM KQML	Knowledge Query and Manipulation Language
LMSs	Learning management systems
LPs	Learning Progressions
LRM	Latent Response Models
LSA	Latent Semantic Analysis
LSCM	Large-Scale Cognitive Modeling
MDPs	Markov decision processes
MICE	Michigan Intelligent Coordination Experiment

MOOCS	massive open online courses
MSF	National Science Foundation
MIT	Massachusetts Institute of Technology
NAS	National Academy Science
NLS	Non-Latent Similarity
NP	Newton’s Playground
NPCs	non-player characters
NRC	The National Research Council
NSF	U.S. National Science Foundation
OLMs	open learner models
ONR	U.S. Office of Naval Research
PAL	Personal Assistant for Learning
PART	Projective Adaptive Resonance Theory Model
PMI	Pointwise Mutual Information
PO	primary object
POKS	Partial Order Knowledge Structures
PSLC	Pittsburgh Science of Learning Center
PUDs	Protocol Data Units
QUAID	Question Understanding AID
PAPI	Personal and Private Information”
PSLC	Pittsburgh Science of Learning Center
PFA	performance factors analysis
R	relevance condition
RDF	resource description format
RML	research modeling language
RMML	research modeling language

R-N	relevance and novelty
RPTEL	Research and Practice in Technology Enhanced Learning
RSS	rich site summary
S	satisfaction condition
SB	springboard
SCORM	Sharable Content Object Reference Model
SCM	Structured Construct Model
SEMILAR	Semantic Similarity
SIMILE	Student Information Models for Intelligent Learning Environments
SMART	Student Modeling Approach for Responsive Tutoring
SOAs	service-oriented architectures
SQL	Structured Query Language
SRL	self-regulated learning
STEM	Science, Technology, Engineering and Mathematics
STTC	Simulation and Training Technology Center
SSA	Social Security Administration
TLA	Training and Learning Architecture
UoM	University of Memphis
URL	uniform resource locator
USAMRAA	U.S. Army Medical Research Acquisition Activity
VBS2	Virtual Battle Space 2
WAS	Word Association Space
WBE	web-based education
WPI	Worcester Polytechnic Institute
WTF	Without Thinking Fastidiously
XML	Extensible markup language

INDEX

Design Recommendations for Intelligent Tutoring Systems

Learner Modeling
Volume 1

Design Recommendations for Intelligent Tutoring Systems explores the impact of intelligent tutoring system design on education and training. Specifically, this volume, "Learner Modeling" examines the fundamentals of learner modeling and identifies best practices, emerging concepts and future needs to promote efficient and effective adaptive tutoring solutions. Design recommendations include current, projected, and emerging capabilities within the Generalized Intelligent Framework for Tutoring (GIFT), a modular, service-oriented architecture developed to promote simplified authoring, reuse, standardization, automated instructional management and analysis of tutoring technologies.

Design Recommendations for Intelligent Tutoring Systems
Team Tutoring
Learner Modeling
Learning Effect Evaluations

About the Editors:

- ***Dr. Robert Sottilare*** *is the Chief Technology Officer at the Army Research Laboratory's SFC Paul Ray Smith Simulation & Training Technology Center. He leads adaptive tutoring research within ARL's Learning in Intelligent Tutoring Environments (LITE) Lab and is a co-creator of the Generalized Intelligent Framework for Tutoring (GIFT).*

- ***Dr. Arthur Graesser*** *is a professor in the Department of Psychology and the Institute of Intelligent Systems at the University of Memphis and is a Senior Research Fellow in the Department of Education at the University of Oxford.*

- ***Dr. Xiangen Hu*** *is a professor in the Department of Psychology at The University of Memphis and visiting professor at Central China Normal University. Dr. Hu received his Master (applied mathematics) from Huazhong University of Science & Technology, Master (Social Sciences) and Ph.D. (Cognitive Sciences) from the University of California, Irvine*

- ***Dr. Heather Holden*** *is an adaptive tutoring researcher at the U.S. Army Research Laboratory's SFC Paul Ray Smith Simulation & Training Technology Center. The focus of her research is adaptive tutoring, technology acceptance, and Human-Computer Interaction (HCI). Dr. Holden is a co-creator of the Generalized Intelligent Framework for Tutoring (GIFT).*

A Volume in the Adaptive Tutoring Series

9 7 8 - 0 - 9 8 9 3 9 2 3 - 0 - 3