Recommendations for Modern Tools to Author Tutoring Systems

Keith Brawner, Heather Holden, Benjamin Goldberg, Robert Sottilare
Army Research Laboratory
Orlando, FL
[Keith.W.Brawner, Heather.K.Holden, Benjamin.S.Goldberg, Robert.Sottilare]@us.army.mil

ABSTRACT

The functional components of any tutor, be it human or computer, can be broken down by process: presenting content to the learner, assessing the performance of the learner, making an instructional strategy decision, implementing this strategy, and determining the impact. While an ideal Intelligent Tutoring System (ITS) can perform all of these actions, the construction of such a system poses its own problem: authoring these functions. This is part of the reason that ITSs are frequently built as monolithic single-topic systems, rather than modular, open-architecture frameworks. An ideal authoring tool would allow a user to create, or have automatically created, all the elements of an ITS with little interaction. While this goal is far reaching, there has been significant effort in the creation and evaluation of tools to support at least one function of tutoring.

The creation of adaptive training is likely to take longer than traditional training for the foreseeable future. This is for one simple reason: more content is required. Due to the corresponding increase in effort, each organization must decide individually whether the performance improvement is worth the cost. However, developmental costs are declining due to the utilization of user tools and automated computer tools to develop the content and function of the intelligent tutor. Research in this area focuses on streamlining the process of authoring adaptive content.

This paper discusses the issues and successes in the development of authoring tools used to generate adaptive training content and functionality. This includes automated tools, such as those for unobtrusively capturing Subject Matter Expert (SME) performance for the purpose of student assessment. This also includes human user tools, such as cognitive model authoring via SME input. The authors conclude with recommendations, based on current research, providing direction to the tasks of creating adaptive content and function.

ABOUT THE AUTHORS

Keith W. Brawner is a researcher for the Learning in Intelligent Tutoring Environments (LITE) Lab within the U. S. Army Research Laboratory's Human Research & Engineering Directorate (ARL-HRED). He has 6 years of experience within U.S. Army and Navy acquisition, development, and research agencies. He holds a Masters degree in Computer Engineering with a focus on Intelligent Systems and Machine Learning from the University of Central Florida, and is currently a PhD Candidate for a doctoral degree in the same field. The focus of his current research is in machine learning, adaptive training, affective computing, datastream mining, and semi/fully automated user tools for adaptive training content.

Heather K. Holden, Ph.D. is a researcher in the Learning in Intelligent Tutoring Environments (LITE) Lab within the U.S. Army Research Laboratory – Human Research & Engineering Directorate (ARL-HRED). The focus of her research is in artificial intelligence and its application to education and training; technology acceptance and Human-Computer Interaction. Dr. Holden's doctoral research evaluated the relationship between teachers' technology acceptance and usage behaviors to better understand the perceived usability and utilization 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 PhD and MS were earned in Information Systems from the University of Maryland, Baltimore County. Dr. Holden also possesses a BS in Computer Science from the University of Maryland, Eastern Shore.

Benjamin S. Goldberg is a member of 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. He has been conducting research in the Modeling and Simulation community for the past 3 years with a focus on adaptive learning and how to leverage Artificial Intelligence tools and methods for adaptive computer-based instruction. Mr. Goldberg is a Ph.D. student at the University of Central Florida and holds an M.S. in Modeling and Simulation. 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. Recent publications include a proceedings paper in the 2010 International Conference on Applied Human Factors and Ergonomics and a poster paper presented at the 2010 Spring Simulation Multiconference.

Robert A. Sottilare, Ph.D. is the Associate Director for Science & Technology within the U.S. Army Research Laboratory - Human Research and Engineering Directorate (ARL-HRED) and directs research within the Learning in Intelligent Tutoring Environments (LITE) Laboratory at ARL's SFC Paul Ray Smith Simulation & Training Technology Center (STTC). He has 28 years of experience as both a U.S. Army and Navy training and simulation researcher, engineer and program manager. He leads the international program at STTC and chairs training technology panels within The Technical Cooperation Program (TTCP) and NATO. Dr. Sottilare holds a patent for a high-resolution, head-mounted projection display and his recent publications have appeared in the Educational Technology Journal, the Journal for Defense Modeling and Simulation and the proceedings of the Intelligent Tutoring Systems Conference 2010. He is a graduate of the Advanced Program Managers Course at the Defense Systems Management College, and his doctorate in modeling & simulation with a focus in intelligent systems is from the University of Central Florida. In January 2012, Dr. Sottilare was honored as the inaugural recipient of the U.S. Army Research Development & Engineering Command's (RDECOM's) Modeling & Simulation Lifetime Achievement Award. The focus of his current research is on the application of artificial intelligence tools and methods to adaptive training environments.

Recommendations for Modern Tools to Author Tutoring Systems

Keith Brawner, Heather Holden, Benjamin Goldberg,, Robert Sottilare
Army Research Laboratory
Orlando, FL
[Keith.W.Brawner, Heather.K.Holden, Benjamin.S. Goldberg, Robert.Sottilare]@us.army.mil

INTRODUCTION

Intelligent Tutoring Systems (ITSs) have the ambitious goal of improving the individual effectiveness of students that they teach, measured through learning gains, content consumption, retention, or other manner. While this goal is shared by all practical systems, each system may have a unique implementation. However, there is a wide body of research which suggests that the construction of an ITS is an appropriate task to undertake. ITSs, on average, produce one effect size, or letter grade, of learning gain (Verdu, Regueras et al. 2008) and this is part of what has driven development in this field. Over the last five years, there has been increased emphasis within the US Army (TRADOC, 2011) on tailored and adaptive training methodologies that include ITS (Army 2011). These ITSs are usually handcrafted, one-of-a-kind systems that are costly and time consuming to develop, and include few standardize elements to promote reuse. The purpose of this paper is to provide recommendations to the creators of intelligent tutoring content and tools about methods to automate authoring, and reduce costs through standard interfaces, processes and components.

The development of a domain-independent, open-architecture, freely-available framework for intelligent tutoring necessitates the development of tools to create the functionality contained within the system. As the functionality in the system grows to accommodate various types of use cases, the tasks of adaptive training creation will eventually be the responsibility of the training coordinator, or instructor. The instructor, while presumed to be a reasonably competent software user, will lack the technical expertise required to design all parts of adaptive training with the addition of software tools. While ideally each of the functions could be automatically run, the reality is that there must be a minimum set of operational components.

Each human tutor, similar to each ITS, must address the fundamental problems inherent in instruction (Beck, Stern et al. 1996). The idea of these components, shown in Figure 1, is not new, and these functions have been agreed upon for a significant time (Barr and Feigenbaum 1982) (A. 1985). Each of these functions must be created, constructed, or authored, and it is traditional for this process to involve a human user. The authoring of these components provides a roadmap for the work left to be done in the area of ITS authoring, and tools required to accomplish such.

The first of these ITS functions is the "training system", which consists of anything that may present content to the student. This 'training system' may be a textbook, a classroom lecture, computer courseware, or flight simulator, but all serve to provide an environment for the student to learn. For our purposes, this is likely to be a pre-existing system or conglomerate of systems which currently train to previously set standards: there are preexisting military textbooks, lectures, etc.

In order for these components to have value to an ITS, there must be some way to evaluate student performance. In the traditional classroom model, this is the "teach, test, teach" or "sound, listen for echoes, sound" loop. An ITS may function in the same manner; it may evaluate as it teaches through exercises, or it may simply observe exercises and help as necessary. In any of these situations, the evaluation of performance is key to responding to it. These evaluations can be qualitative or quantitative (Ekanayake, Backlund et al. 2011). The chokepoint in the development stream is the authoring of these qualities.

There is an open research question about what types of intervention actions to take when performance is assessed. Questions such as "*is the learner ready to move on to new concepts?*", "*how much should the system solve for the learner?*" and "*what type of*

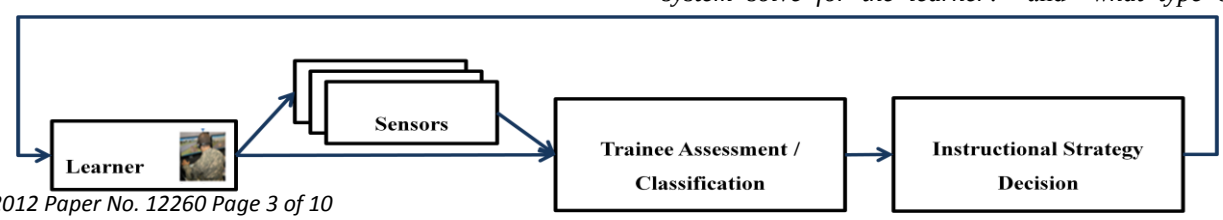

Figure 1 – Basic Operational Loop of Intelligent Tutoring (Sensors optional)

feedback is ultimately constructive for this learner?" arise whether performance is assessed positively or negatively. The instructional strategy selection engine, whether it is composed via if/then programmatic statements, or decision tree logic, must be constructed.

However, each of these types of interventions must trace back to an action which the system is able to assign. After an instructional strategy (give a hint, for example) is selected, this must eventually be related to a hint which the training system is able to give. Many times, this network must be created by hand, but new efforts in the area of automatic hinting may prove useful for many types of content-specific instruction.

In the below sections of this paper, the authoring of each of these adaptive tutoring functions (shown in Figure 2) will be discussed, with recommendations presented on the best paths forward. Specifically, these sections are related to content presentation, student assessment, pedagogical strategy selection, and the responses to these strategies (the actual hints/prompts/pumps/etc.). Finally, the reader will be presented with recommendations towards the development and use of various tools to adapt training content.

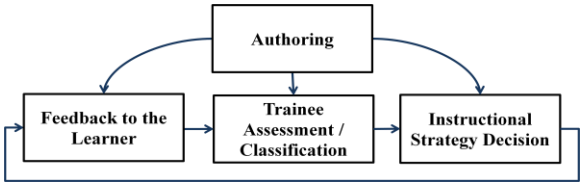

Figure 2 - Authoring the components of tutoring

CREATING CONTENT

There is a vast amount of content for organizational learning which has already been created. The capture, storage, and retrieval of information in a machine-readable format, such as an ontology, is useful to organize and redistribute this knowledge to the users who require it. Inferences among ontological nodes can then be used to make inferences on other nodes, meet the needs of learners, and be easily redistributed (Noy and McGuinness 2001).

While content is plentiful, the need for variations in content (e.g., branching scenarios) is growing with the need for adaptive systems (e.g., computer-based tutoring) where tailored instruction requires different content for each individual depending on their competency, preferences and other individual differences. This requirement for "on demand" content means adaptive instruction will become unaffordable unless methods are devised to automate content creation and promote standards for reuse.

Research prototypes such as the Army's Soldier-Centered Army Learning Environment (SCALE) allow for a user to customize a course through the selection of instructional ontological nodes, although this content is still currently hard-written. These nodes, in turn, can be generated by the instructor and/or through automated processes for knowledge ingestion from the semantic web, Sharable Content Object Reference Model (Bohl, Scheuhase et al. 2002), or web-based reference systems (Jesukiewicz and Rehak 2011). These standards enable the creation of powerful course-creating tools, allow for the upload of content developed through crowdsourcing (distributed development), and promote reuse, thereby enhancing the performance of instructional designers, training system developers, and trainers/teachers.

The compatibility and interactivity of media are limiting factors in using existing content objects. Interactive multimedia instruction (IMI) defines four levels of interactivity (Schwier and Misanchuk 1993). Existing content objects range from low interactivity (e.g., information presentations) to moderate interactivity (e.g., desktop game-based training) to high interactivity (e.g., fully immersive three-dimension simulation environments). Standards are needed to allow for easy integration of existing content into frameworks (e.g., ARL's Generalized Intelligent Framework for Tutoring) that include pedagogical engines to deliver tailored training experiences.

As important as methods for content creation are, they are less than optimal without metadata to support machine-level understanding of ontological features: objects, classes, attributes, relationships, and most importantly source information. Several standards exist for ontology languages and include, but are not limited to: Common Logic (Delugach, 2006), IDEF5 (Benjamin, Menzel et al. 1994), and OWL (Sirin, Parsia et al. 2007). OWL developed by the Web3D consortium is widely used and may be a defacto standard for training ontologies in the future.

CREATING LEARNER MODELS

To date, there are no standardized methods for creating learner models. Earlier ITSs primarily modeled students' answer choices and

misconceptions (i.e., bugs or errors) as representations of a student's knowledge at any given point of time. The earliest ITS implementations, within the 1970s commonly used overlay learner modeling, a method which assumes the student's knowledge is a subset of the correct domain knowledge, as outlined within an expert model. Although this type of modeling is simplistic in nature, overlay models do not have the capability to capture misconceptions within a student's knowledge (Burton 1982). Thus, ITSs of the 1980s began to use bug libraries, which are built to match students' behavior, for capturing misconceptions. These libraries (catalogs) could not encompass the wide range of student behaviors and were difficult to construct. Ohlsson and Langley (1985) attempted to use induction as a method to create a learner model from previous examples of student behavior (Ohlsson and Langley 1985). While providing more flexibility than bug libraries, the induction learner modeling method could not correctly accommodate preprogrammed misconceptions (Baffes and Mooney 1992).

In the 1990s, researchers began looking to machine learning techniques for learner modeling. One of the learner modeling systems using such techniques is called ASSERT (Acquiring Stereotypical Student Errors using Revision of Theories). This system performs learner modeling and automatically generates libraries of common bugs with a general purpose algorithm that uses domain-independent machine learning techniques. ASSERT constructs its bug libraries by extracting commonalities across multiple models (Baffes and Mooney 1992). Researchers also began to consider the importance of accounting for learners' individual characteristics. An Authoring Tool for User Model Management (ATUMM) is a system that will allow an author to create a Learner Model Manager for developing a learner model. ATUMM was only developed as a prototype system; however, it acknowledged that learner models should contain a history of what the learner has done, attributes that reflect the learner's permanent and transient characteristics, and both behavioral and conceptual knowledge of the learners' beliefs towards the domain (Self 1991).

Around the mid-1990s, researchers began developing authoring tools for developing ITSs to help make ITS knowledge and functional components reusable and sharable between systems. With a basis of ontology engineering, several implementations used ontological approaches for generating learner models. One ontology-based authoring tool, called SmartTrainer Authoring tool, has used a learner model ontology that included axioms as guidelines for constructing learner modeling systems. The author is able to construct the learner model according to the target knowledge of teaching (i.e., declarative, procedural, etc.) (Chen, Hayashi et al. 1998). Another example of an ontological approach to designing learner models was created for a Tutorial Agent System (TAS). This model identified the underlying reasons for students' systematic and predictable misconceptions using a knowledge representation scheme called InfoMap (Tu, Hsu et al. 2002).

The notion behind these ontology approaches is that they are task-independent and domain-independent; however, each of the existing prototypes mentioned above have only been used within one exemplar domain area. Although ontology approaches contain independent knowledge representation, such methods are incapable of accurately modeling interdependencies of knowledge. Thus, the most relevant wave of learner modeling construction has utilized dynamic belief networks (Reye 1999). These networks provide a systematic approach to gathering information about the scope of a student's knowledge and can account for interdependent relationships within such knowledge. Researchers have employed the Belief Net Backbone/structure (Reye 1996; Reye 1999; Reye 2004) to generate Bayesian learner models (Zapata-Rivera and Greer 2000). Zapata-Rivera and Greer then developed learner models that were then used to develop two tools: ViSMod, an interactive visualization tool for Bayesian learner models, and ConceptLab, a navigation and knowledge construction system that uses XML-based conceptual maps for representation of the learner's view of the domain (Zapata-Rivera and Greer 2001). These tools allowed for the collaborative construction and inspection of learner models by various users (i.e., students, teachers, instructional designers, etc.). Open learner models are shown to improve the quality of learner models and fostering reflective thinking (Dimitrova 2003).

There are many other methods for developing learner models for domain-independency, such as the Cognitive-Trait Model (CTM) (Lin, Kinshuk et al. 2007). However, within the past decade, ITS researchers have started to incorporate affective modeling into the development of learner models. Traditionally, affect was not considered part of the cognitive process; however, research has shown that both cognition and affect are intricately intertwined (Picard, Papert et al. 2004). Affective learner modeling, in addition to cognitive modeling techniques, can allow for better diagnosis and

response of a student's state of knowledge at any given time. Thus, future authoring tools for creating learner models should provide the appropriate capabilities to model both types of states in addition to the learner's individual attributes for optimal interpretation of the learner's state. Such systems should be modular, domain-independent, and easily transferrable between different ITSs.

CREATING EXPERT MODELS

Current tutoring systems have a tendency to use heuristics, or "rules of thumb" in order to assess students. These types of rules typically remain in place, as they were written, as the content for the system expands (Lesta and Yacef 2002). The classical example of this is the educational system outgrowth from the field of expert systems and knowledge engineering (Morrison, Kobus et al. 2006). In this type of system, the methods for student assessment are created singularly, by hand, and typically by a programmer. Advances in this field allowed for experts to create their own rules, although these still required a check for consistency by a computer system, or developer (Muldner, Burleson et al. 2010).

The next generation of tutors is imagined to make use of dynamic models which can infer hidden learner characteristics, and recognized unanticipated behavior based on learner performance, past experiences, and lessons learned. These systems include game-based training environments, interactive features, and virtual worlds, where content is developed continuously and separately from the environment (Nkambou 2006). Methods for the hand-development of expert models are labor-intensive, require large amounts of expert time, and frequently require expert understanding of the underlying technology. Two examples of these systems which have left the laboratory and are in practical use are the APSPIRE (Mitrovic, Suraweera et al. 2006) and Cognitive Tutor Authoring Tools (CTAT) systems (Aleven and Koedinger 2002), which have experimentally enabled 200 hours of content creation to 1 hour of content presentation (200:1) and 40:1 authoring (Heffernan, Turner et al. 2006), respectively.

There is some amount of hope that the development of expert models can be automated, rather than hand-authored. The development of expert models from the observation of underlying data interactions is not necessarily a novel idea (Nwana 1990), but has not seen widespread dissemination because of the single-domain focus of existing systems. If we are to provide recommendations for the creation of expert models, they should follow the lines of automatic creation from the data. As such, it is desirable for expert models to be created automatically, either from the data which is interacted with, or from the data of interaction.

The first method in the performance of automatic expert model creation is in the analysis of traditional training material. In the absence of a three dimensional training game, there is frequently the presence of PowerPoint documents, Word documents, policy documents, or other compositions of a domain corpus. These can be analyzed to form competency clusters of documents describing similar tasks which have prerequisite knowledge towards other competency clusters. The further use of semantic search may be able to discover inherent task descriptions. These are the critical components which aid in the creation of model-/constraint-tracing approaches to assessment (Olney, Graesser et al. 2010).

The other method of automated, domain independent, expert model creation relies on the inherent interactions that a Subject Matter Expert (SME) has with the system. Collections of AI methods such as Reinforcement Learning (RL) are able to mimic expert performance through repeated presentation of input/output pairs observed from experts. This type of technology should easily expand to the observation of expert system interactions for the classification of novice system interactions. In this manner, one can assess the particular rule/node/input relation that the novice has violated, and attempt to remediate or reinforce performance. The "train the system to train others" approach does not require the expert to go through extensive knowledge elicitation, which has long been an issue (Ok-choon, Ray et al. 1987).

Both of these methods can be applied in a manner which skirts the problem of creating a single-domain system. These approaches can be applied across very different types of training tasks, and aid significantly in reusability. It is the hope that this recommendation enables others to create rules for assessment in significantly reduced times across multiple arenas of training.

CREATING INSTRUCTIONAL STRATEGIES

The future effectiveness of ITSs lies in the capability of accurately monitoring performance states in real-time and applying instructional strategies tailored to the individual user. Theory dating back to 2003 has stressed the differences between systems which know

how to teach, rather than what to teach, and have stressed the need to authoring of these particular functions (Murray 2003). These tools assist in the authoring of pedagogy based around individual differences of the user with the intent to be implemented in computer-based training. To lessen the knowledge base required for the formation of ITS enabled training, the creation of instructional strategy components should follow a standardized domain-independent process that assists content authors in applying empirically-proven strategies. The goal is for the development of authoring tools that guide trainers in identifying proven strategies appropriate for the domain being trained along with guidance for authoring specific functions to be implemented. Commonly applied strategies are derived from research on techniques and tactics employed by expert tutors in a one-on-one learning environment found to improve performance outcomes (Boulay and Luckin 2001; Person and Graesser 2003). To this effect, instructional components are tailored prior to interaction to better suit a user's ability within a given domain, and guidance and adaptation is facilitated in real-time based on monitored system interactions. These functions expand beyond pedagogical approaches implemented in previously developed ITSs that solely use feedback in response to error (Anderson, Boyle et al. 1987; Mason and Bruning 2001).

Integrating pedagogical function into an ITS requires a functional infrastructure. First, performance assessments are required that highlight deficiencies and errors among associated training objectives. Interaction data within a training platform must be interpreted and linked to specific objectives that designate desired performance. Understanding the root cause of an error, down to the specific decision or misconception, is essential when selecting a strategy to execute. With this information, an ITS can focus on the knowledge components associated with a diagnosed deficiency. Furthermore, modern ITSs look at reactive states during interaction as a source for adaptation. These include cognitive and affective variables found to impact learning outcomes (i.e., attention, workload, emotions, mood, etc.). Physiological and behavioral metrics associated with a user provide insight into an individual's "readiness to learn" and can inform adaptation to mediate negative states. As well, monitoring this information in real-time can provide further context into classified errors. For example, when observing cognitive state for two individual's with low performance assessments, one can be credited to boredom while the other is due to high cognitive load. This information drives the application of different adaptations to difficulty level.

With defined triggers of feedback and adaptation, information pertaining to the user and training content will drive the specific strategies and tactics to apply. This includes individual differences that have been found to impact learning (i.e., affective traits, cognitive ability, prior knowledge/experience, etc.) and task characteristics pertaining to the objective being tracked (i.e., knowledge type, task difficulty, training environment, etc.). Combining knowledge about the specific trigger of an intervention with knowledge of the specific user, strategies can be tailored to the strengths and weaknesses of the learner.

The next component to reflect on in instructional strategy creation is how an ITS's function is limited by the environment within which it is integrated. The available interfaces and avenues for applying adaptations must be recognized upfront for determining available generalized strategies. This requires understanding how information communicated from the tutor can be displayed and the available environmental components that can be manipulated. With mechanisms in place for determining when an intervention is deemed appropriate and tools for executing the intervention within the training environment, specific strategies must be authored and managed.

For an authoring tool to generate individualized pedagogy, AI methods must be explored for optimized strategy selection. However, this is a problem which has been reasonably well-addressed in the literature (Du and Zhan 2002). Current authoring tools exist, but they have been criticized for having a shallow domain knowledge representation through canned text and graphics (Weyhrauch 1997). To improve the utility of an instructional strategy authoring tool, experimentation and empirical evaluations will be required to determine the optimal set of strategies and modeling approaches. To maintain domain-independency, the developed tool needs to support authoring strategies across variations in ITS models. This must be carried out across multiple problem spaces and computer-based training environments. Structuring sources of adaptation selection with targeted strategies in a self-executing decision tree will provide a basis for personalized authoring, shown in Figure 3. This will assist in defining and testing strategies associated with all facets of training.

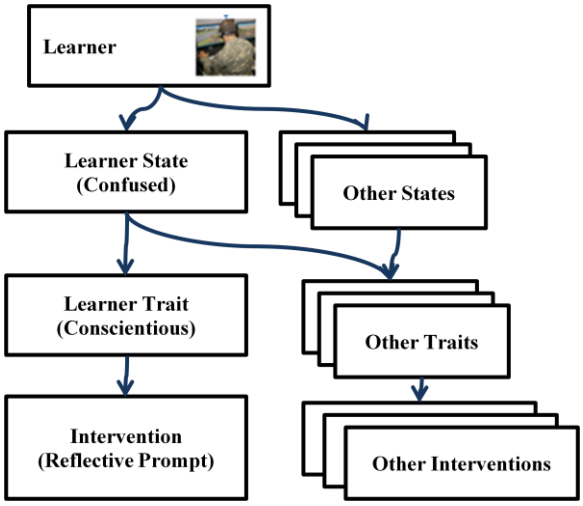

Figure 3 - Example of a decision tree process

CONCLUSION/RECOMMENDATIONS

The greatest strength of intelligent tutoring authoring tools is the ability to achieve domain independence. The use of one authoring tool to author content, assessment, user models, or instruction across multiple types of training is the strongest point for which the authors can argue. This was not historically relevant, as there was not a tutoring system which has also attempted to strive for this benchmark. There are now several multi-domain systems which may support the development of such authoring tools, such as AutoTutor (Susarla, Adcock et al. 2003), Cognitive Tutor (Aleven, McLaren et al. 2006), and the Generalized Intelligent Framework for Tutoring GIFT.

Logically, authoring tools are among the last phase for development of training systems. Just as the developer does not design a scenario editor without support for various exercise activities, a tutor developer does not design an authoring tool until the functional components of the tool have been standardized and practiced. Authoring tools come in various forms, and a sample of some of the 'strong' activities that are practiced includes: WYSIWYG editing, rule generation/checking, micro-/macro-level tutoring strategies, deep models of domain knowledge, predefined concept ontologies, and template-based learning delivery (Murray 1999).

If the reader is to take away the most salient points of this work in their creation of authoring tools, they are that the tool should:

- Be independent of the content domain
- Provide integrated user interfaces supporting development

For content creation, these two points mean that the tools for creating content nodes must subscribe to standards. For the creation of user assessment models, this means that the model should be created from the underlying data or interactions, rather than painstakingly hand-crafted. For the creation of user models they should be related to capturing the expertise and the assignment of individual learner traits and states. For the creation of instructional strategies, this means that the strategy engine should be static, and the user must only author the responses. The overarching goal of a system which is able to tutor multiple areas of training in multiple ways necessitates the drive towards tools which are generic enough to be portable, and integrated enough to be functional.

REFERENCES

A., B. (1985). Artificial Intelligence: Promise and Performance. London, Prentice Hall.

Aleven, V., B. McLaren, et al. (2006). The cognitive tutor authoring tools (ctat): Preliminary evaluation of efficiency gains, Springer.

Aleven, V. A. and K. R. Koedinger (2002). "An effective metacognitive strategy: Learning by doing and explaining with a computer-based Cognitive Tutor." Cognitive Science **26**(2): 147-179.

Anderson, J. R., C. F. Boyle, et al. (1987). Cognitive Principles in the Design of Computer Tutors. Modelling Cognition. P. Morris, John Wiley & Sons Ltd.**:** 93-133.

Army, D. o. t. (2011). The U.S. Army Learning Concept for 2015. TRADOC.

Baffes, P. T. and R. J. Mooney (1992). Using Theory Revision to Model Students and Acquire Stereotypical Errors. 14th Annual Conference of the Cognitive Science Society, Bloomington, IN.

Barr, A. and E. A. Feigenbaum (1982). The Handbook of Artificial Intelligence. Los Altos, CA, Kaufmann.

Beck, J., M. Stern, et al. (1996). "Applications of AI in Education." ACM Crossroads **3**(1): 11-15.

Benjamin, P. C., C. Menzel, et al. (1994). "Idef5 method report." Knowledge Based Systems, Inc.

Bohl, O., J. Scheuhase, et al. (2002). The sharable content object reference model (SCORM)-a critical review, IEEE.

Boulay, B. d. and R. Luckin (2001). "Modelling Human Teaching Tactics and Strategies for Tutoring Systems." International Journal of Artificial Intelligence in Education **12**: 235-256.

Burton, R. R. (1982). Diagnosing Bugs in a Simple Procedural Skill. Intelligent Tutoring Systems. D. H. Sleeman and J. S. Brown. London, Academic Press.

Chen, W., Y. Hayashi, et al. (1998). An Ontology-Based Intelligent Authoring Tool. 6th International Conference on Computers in Education.

Dimitrova, V. (2003). "STyLE-OLM: Interactive Open Learner Modelling." International Journal of Artificial Intelligence in Education **13**: 35-78.

Du, W. and Z. Zhan (2002). "Building decision tree classifier on private data."

Ekanayake, H., P. Backlund, et al. (2011). Assessing Performance Competence in Training Games. Proceedings of the 4th International Conference on Affective Computing and Intelligent Interaction (ACII 2011), LNCS. S. D. Mello, A. Graesser, B. Schuller and J.-C. Martin. Berlin Heidelberg, Springer-Verlag. **6975:** 518-527.

Heffernan, N. T., T. E. Turner, et al. (2006). The ASSISTment builder: Towards an analysis of cost effectiveness of ITS creation.

Jesukiewicz, P. and D. R. Rehak (2011). The Learning Registry: Sharing Federal Learning Resources, NTSA.

Lesta, L. and K. Yacef (2002). An Intelligent Teaching-Assistant System for Logic. Proceedings of Intelligent Tutoring Systems. S. Cerri and F. Paraguo. Biarritz, France.

Lin, T., Kinshuk, et al. (2007). Cognitive Trait Model and Divergent Associative Learning. IEEE International Conference on Advance Learning Technologies (ICALT 2007), Niigaga, Japan, IEEE.

Mason, B. J. and R. Bruning (2001). Providing Feedback in Computer-Based Instruction: What the Research Tells Us. University of Nebraska-Lincoln, Center of Instructional Innovation.

Mitrovic, A., P. Suraweera, et al. (2006). Authoring constraint-based tutors in ASPIRE, Springer.

Morrison, J. G., D. A. Kobus, et al. (2006). DARPA Improving Warfighter Information Intake Under Stress –Augmented Cognition, Phase II: The Concept Validation Experiment (Technical Report 1940). San Diego, California, SPAWAR Systems Center.

Muldner, K., W. Burleson, et al. (2010). "Yes!": Using Tutor and Sensor Data to Predict Moments of delight during Instructional Activities. User Modeling, Adaptation, and Personalization: proceedings of 18th International Conference, UMAP 2010. P. D. Bra, A. Kobsa and D. Chin. Berlin, Springer.

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

Murray, T. (2003). "An overview of intelligent tutoring system authoring tools." Authoring tools for advanced technology learning environments: 491-544.

Nkambou, R. (2006). Towards Affective Intelligent Tutoring System, Workshop on Motivational and Affective Issues in ITS. 8th International Conference on ITS**:** 5-12.

Noy, N. F. and D. L. McGuinness (2001). Ontology development 101: A guide to creating your first ontology, Stanford knowledge systems laboratory technical report KSL-01-05 and Stanford medical informatics technical report SMI-2001-0880.

Nwana, H. S. (1990). "Intelligent Tutoring Systems: An Overview." Artificial Intelligence Review **4**(4).

Ohlsson, S. and P. Langley (1985). Identifying Solution Paths in Cognitive Diagnosis. Technical Report CMU-RI-TR-85-2. Pittsburg, PA, Carnegie-Mellon University.

Ok-choon, P., S. P. Ray, et al. (1987). Intelligent CAI: old wine in new bottles or a new vintage? Artificial Intelligence and Instruction: Instruction and Methods. Reading, MA, Addison-Wesley**:** 11-43.

Olney, A., A. C. Graesser, et al. (2010). Tutorial Dialog in Natural Language. Advances in Intelligent Tutoring Systems, Studies in Computational Intelligence. R. Nkambou, J. Bourdeau and R. Mizoguchi. Berlin, Springer-Verlag. **308:** 181-206.

Person, N. K. and A. C. Graesser (2003). Fourteen Facts about Human Tutoring: Food for Thought for ITS Developers. Artificial Intelligence in Education 2003 Workshop Proceedings on Tutorial Dialogue Systems: With a View Toward the Classroom V. Aleven, U. Hoppe, J. Kayet al. Sydney, Australia 335-344.

Picard, R., S. Papert, et al. (2004). "Affective Learning -- A Manifesto." BT Technology Journal **22**(4): 253-269.

Reye, J. (1996). A Belief Net Backbone for Student Modelling. International Conference on Intelligent Tutoring Systems. C. Frasson, C. Gauthier and A. Lesgold. Montreal, Canada, Springer-Verlag**:** 595-604.

Reye, J. (1999). "Student Modelling based on Belief Networks." International Journal of Artificial Intelligence in Education **11**.

Reye, J. (2004). "Student Modeling based on Belief Networks." International Journal of Artificial Intelligence in Education **14**: 1-33.

Schwier, R. and E. R. Misanchuk (1993). Interactive multimedia instruction, Educational Technology Pubns.

Self, J. (1991). "ATUMM: an Authoring Tool for User Model Management." Interactive Journal International **7**: 277-292.

Sirin, E., B. Parsia, et al. (2007). "Pellet: A practical owl-dl reasoner." Web Semantics: science, services and agents on the World Wide Web **5**(2): 51-53.

Susarla, S., A. Adcock, et al. (2003). Development and evaluation of a lesson authoring tool for AutoTutor.

Tu, L. Y., W. L. Hsu, et al. (2002). A Cognitive Student Model--An Ontological Approach. International Conference on Computers in Education, Washington, DC, IEEE Computer Society.

Verdu, E., L. M. Regueras, et al. (2008). "Is Adaptive Learning Effective? A Review of the Research." Wseas Advances on Applied Computer and Applied Computational Science: 710-715.

Weyhrauch, P. (1997). Guiding Interactive Drama. , Carnegie Mellon University. **Ph.D**.

Zapata-Rivera, J.-D. and J. Greer (2000). Inspecting and Visualizing Distributed Bayesian Student Models. 5th International Conference on Intelligent Tutoring Systems, Springer-Verlag London, UK.

Zapata-Rivera, J.-D. and J. Greer (2001). Externalising Learner Modeling Representations. AI-ED 2001 Workshop: External Representations in AIED: Multiple Forms and Multiple Roles. San Antoni, TX.