# Proposing Module-level Interoperability for Adaptive Instructional Systems

Keith Brawner and Robert Sottilare

Army Research Laboratory
{keith.w.brawner.civ;robert.a.sottilare.civ}@mail.mil

**Abstract.** The core components of a tutoring systems appear to be widely agreed upon. As an example, Murray's seminal review work from 1999 included the components of the student interface, domain model, teaching model, and student model [1]. Woolf's 2011 review of ITS divides the space into models of student knowledge, domain knowledge, tutoring knowledge and the communication of knowledge [2]. VanLehn's review work describing the behavior of tutoring systems mentions "that although tutoring systems differ widely in their task domains, user interfaces, software structures, knowledge bases, etc., their behaviors are in fact quite similar", clearly viewed the world through a lens of finite modeling paradigms [3]. The author has worked significantly on the Generalized Intelligent Framework for Tutoring (GIFT) system, which includes models of a domain, pedagogical, learner, and external interface explicitly. It seems clear that each Intelligent Tutoring System (ITS) must, at a minimum, include a model of the domain of instruction, a model of the learner that it is instructing, and a model of instruction to deliver. This paper discusses component-level standardization for data interchange as well as proposing that the base components are the one which have been widely agreed upon.

**Keywords:** Standards, Interoperability, Adaptive Instruction, Intelligent Tutoring Systems

## 1 The interchangeable parts of an ~~ITS~~ AIS

The abstract discusses the basic, widely-known, parts of a standard ITS; models of the domain, leaner, pedagogy, and interface. Naturally, these models can be as simple of a set as "one algebra problem" (domain model), "Right/Wrong" (learner model), and "Do it until you get it right" (instructional model). This can be as complicated as an interactive scenario with varying levels of difficulty and feedback (domain model), a lifelong history of all learner activities with assessment of competencies (learner model), and a system of feedback timing and difficulty adjustment (instructional model). While these core components of an ITS are somewhat agreed upon, few organizations outside of the Army Research Laboratory have constructed greater than 50 ITSs from a common core of components [4]. This would be a great engineering feat if not for the use of interchangeable components.

Interchangeable components allow for a few significant goals to be accomplished. The first of these is the fair evaluation of one component against another. For individual learners, is a system of Socratic dialogue (ala AutoTutor) or direct immediate feedback (ala Cognitive Tutor) the most appropriate action to take? For low motivated learners? For prior domain experts? These interesting research questions are enabled via interchangeable components. It is not possible to answer these types of research questions without crafting a whole system, at the moment. The second of these goals is to provide a common basis on which to construct authoring tools. A set of authoring tools which can construct many tutors obviates the need for both tutor construction and component construction.

### 1.1 Adaptive Instructional Systems (AISs)

The term "Adaptive Instructional System" (AIS) encompasses many of the different types of systems which exist within the field. It encompasses, but is not limited to, the ITS family of systems. The authors are proposing the use of the AIS term to help to define standards among systems which interchange the same types of information - a pedagogical engine is not an ITS, but is a part of an AIS; an adaptive test is not an ITS, but can conform to AIS standards; a indexed content repository with various metadata is a component of an AIS and should be compliant with standards. The AIS term is broad enough to encompass all systems with a stated goal of enabling both a) adaption, and b) instruction, based on the learner.

Previous work has discussed basing design decisions around the idea of the Lowest Replaceable Unit (LRU), the lowest level of compliance [5]. This idea is roughly equivalent to ideas in other fields, such as the "Unit" from software engineering, an item which can be independently tested at the input/output level for conformance to desired behavior, in the software standards community [6]. Similarly, in the early business stage software community, they describe the Minimum Viable Product (MVP) as the smallest unit which produces a market-acceptable service (i.e., "scratches an itch"). AISs are composed of many of these base-level components, and although there is debate among what the base level is, its description is not – it is the minimum testable and viable software module or service.

## 2 Case Study: Interchangeable Instructional Models and Modules

As the GIFT system has been developed, it has started with simplistic models and experimented with more complex models in each of the model categories. As an example, the initial models served merely as "pass-through" models – the Learner Module forwarded all of its information to the Pedagogical Module without meaningful processing. Later, the learner model was reaching out to external Learner Record Stores (LRS) [7] systems in order to aggregate learner performance, and recommending con-

tent in an instructionally simplistic manner [8]. In the current GIFT release, this information is used in more complex manners, including the injection of content mid-lesson [9].

The GIFT production system has additionally changed over time. The initial instructional model, like the Learner Module, consisted of "pass-through" functionality. This was quickly exchanged for a model which used information about the learner in order to drive content selections, while still operating in an error-sensitive feedback manner within simulated environments [10]. This 2012 model, based on Component Display Theory[11], has since been replaced with a model which performs similar actions, but tracks its past actions over time in order to learn which content types of an individual learner/course make the largest difference [12], based on the Interactive/Constructive/Active/Passive remediation framework [13]. Because of the nature of interchangeable parts, each of these models is still usable in the current system – a system designer can choose among these models to use without making *any* changes to the models of the learner, models of the content, or models of delivery. The internal evidence is suggesting that the reinforcement-based models are having the highest performance *in situ*, and have been set to the default model types for newly created content.

The ability to interchange models has significant advantages. It allows for the direct comparison of models. As an example, such a system allows for the answering of the question: "What would the effect be if the Cognitive Tutor were to ask introspective questions in the manner of AutoTutor instead of selecting problems?" This allows for the selection of a "best of breed" model for each of the tasks. Next, it enables significant levels of reuse – one model can be used for many systems. Finally, it allows for the creation of authoring tools which can author content, independent of the other models - an instructional model can be constructed without knowledge of the domain model and hence be useful for a wide variety of training applications; a domain model can be created with standard tools without particular concern about the motivation levels or prior experience of the learners (handled by the learner model and instructional model, respectively). The abilities to compare, reuse, author, and encapsulate complexity drive down overall cost and development time.

## 3 Where to Start?

GIFT adopted the design principle of separating content from executable code [14]. This is more than simply the separation of domain content from the instructional content – but multiple layers of separation of logic. Modules (operational software processes) are separated from the data they process in either through the use of configuration information (i.e., domain content, sensor configuration settings), restrictions on the types of data input and output (i.e., interface specifications), or software library calls (i.e., an outside call to a library to provide assessment of student information). This provided a starting point of the GIFT system and the authors believe that the evolving messages and standards provide a natural starting point for standardization.

We recommend an analysis of the information required by AIS common components to begin formulating an ontology as a basis for selecting candidates for standard messages. Given that there are a number of relatively finite components, the natural question is "where do we start?" Based on experience with GIFT and in reviewing the literature, Table 1 lists some of the most frequently recurring examples, but is intended to be illustrative, not exhaustive. Table 1 provides a starting place for component-level standardization discussion, and invites comment. This initial table of items provides the starting point from which GIFT has built upon.

**Table 1.** Table of proposed messages for initial module-level interoperability

| |
|---|
| Domain Model |
| Input |
| Requests for action (from Instructional Model) |
| Feedback associated with concepts (optional field) |
| A model of tasks and conditions, so as to generate Output |
| Output |
| Learner Assessment (to Learner Model) |
| Learner Model |
| Input |
| Learner assessments for each learning objective or concept (from Domain Model) |
| Sensor data (if applicable) |
| Longer term data (if applicable) |
| Output |
| Learner State representation (from Domain Model or derived from data) |
| Pedagogical Model |
| Input |
| Learning State representation (from Learner Model) |
| Cognitive state of the learner (optional) |
| Performance expectations (above, below, at) for each concept |
| Predicted future performance based on competency model (optional) |
| Physiological State representation (from Learner Model) |
| Derived emotional, physical states (e.g., fatigue) (optional) |
| Physiological stressors (optional) |
| Behavioral State representation (from Learner Model) |
| Derived attitudes or psychomotor performance based on primitive behaviors (optional) |
| Longer term learner attributes (from Learner Model or LRS) |
| Demographics and traits (optional) |
| Historical performance (competency) (optional) |
| Output (all optional) |
| Request for course direction |
| Request for feedback |
| Request for scenario adaption |
| Request for assessment |

### 3.1 Case Study Example: Hinting amongst AutoTutor, Cognitive Tutor, and GIFT

As an example, consider the behavior of somewhat disparate systems – AutoTutor, Cognitive Tutor, and GIFT. Each of them provides very different models of the student, instruction, domain, and other items. However, at the base level, each of the systems has a "last piece of feedback" item built into the system. In Cognitive Tutor, this is called the "bottom out" hint within its hinting model [15]. In AutoTutor, if the student has not responded to Socratic question asking over the course of the tutorial session, the student is given a hint, prompt, pump, and eventually the "assertion" [16]. In GIFT, its default behavior is to repeatedly give the final hint within a hint sequence [17]. The standard representation, presented above, allows for each of the systems AutoTutor to output the final level of feedback (bottom out, assertion, repeated hinting, respectively) as part of a unified standard. While these systems are different in their design, intent, and domains of instruction, their actual behavior can be well-represented within standardized communication protocols.

## 4 Discussions

A typical "shell tutor" can be used to create a significant amount of content which conforms to a relatively finite number of templates, such as "the vast majority of mathematics problems" in the case of the Cognitive Tutor. GIFT, however, is a tutoring architecture that supports the authoring of ITSs, the delivery and management of adaptive instruction, and the evaluation of ITSs and adaptive instructional capabilities for nearly any task domain (e.g., cognitive, affective, psychomotor, social). It can be, and has been, used to tutor or adjust on the above described uses cases – simulations, content, and experiments. It has done so through the engineering of interchangeable parts and standardized interfaces. The authors believe that these interfaces can provide a starting point for other ITSs, and have previously worked to communicate domain information with AutoTutor, Cognitive Tutor, and Betty's Brain systems; there is reasonable confidence that these interfaces are sufficient to enable relatively advanced tutoring techniques.

### 4.1 Market Opportunities Created

The current business practice, with a lack of standards, forces the monolithic creation of a single AIS system. While this single AIS system may make use of a variety of standards (i.e., JSON, XML, TCP/IP, LTI reporting standards, etc.), the end result is that its components are not interchangeable. The creation of standards enables new business practices and market opportunities.

The basic market opportunities that standardization at this level creates are, roughly, one business model per module of standardization. In this manner, a business case for a "vendor-supplied" instructional engine is created. The business for the learner model

is mostly as an aggregation service with an emphasis on traits relevant to learning, rather than the traditional model of consumer preferences. The business for the domain model is in the sale of content which can train individuals for their tasks. Further, the creation of individual models creates new business models which service the needs of the businesses based on the previous ones – content aggregators for learner model data provisioning, system-creators for selection of the appropriate types of models relevant to a task domain, certification businesses for certification that the models interchange as specified, analytic services for existing data reprocessing, etc. If the reader is curious on the kinds of educational businesses which may come into existence around a standard, they need only consider the number of vendors which make use of SCORM [18] for content, or xAPI for learner modeling information logging [7].

## References


1. Murray, T., *Authoring intelligent tutoring systems: An analysis of the state of the art.* International Journal of Artificial Intelligence in Education (IJAIED), 1999. **10**: p. 98-129.
2. Woolf, B., *Intelligent Tutors: Past, Present, and Future.* Keynote address at the Advanced Distributed Learning ImplementationFest, 2011.
3. VanLehn, K., *The behavior of tutoring systems.* International Journal of Artificial Intelligence in Education, 2006. **16**.
4. Brawner, K., A.M. Sinatra, and R. Sottilare, *Motivation and research in architectural intelligent tutoring.* International Journal of Simulation and Process Modelling, 2017. **12**(3-4): p. 300-312.
5. Sottilare, B. and K.W. Brawner, *Exploring Standardization Opportunities by Examining Interaction between Common Adaptive Instructional System Components*, in *Adaptive Instructional Systems Workshop*, B. Sottilare, Editor. 2018, IEEE: Orlando, FL.
6. Committee, I.S., *Standard Glossary of Software Engineering Terminology*, in *IEEE STD 610.12-1990 IEEE*. 1990, IEEE: IEEE.
7. Regan, D.A. *The Training and Learning Architecture: Infrastructure for the Future of Learning*. in *Invited Keynote International Symposium on Information Technology and Communication in Education (SINTICE), Madrid, Spain*. 2013.
8. Brawner, K. and S. Ososky. *The GIFT 2015 Report Card and the State of the Project*. in *Generalized Intelligent Framework for Tutoring (GIFT) Users Symposium (GIFTSym3)*. 2015. Orlando, FL.
9. Folsom-Kovarik, J.T. and M.W. Boyce. *Developing a Pattern Recognition Structure to Tailor Mid-Lesson Feedback*. in *Proceedings of the 5th Annual Generalized Intelligent Framework for Tutoring (GIFT) Users Symposium (GIFTSym5)*. 2017. Robert Sottilare.
10. Goldberg, B., et al. *Use of Evidence-based Strategies to Enhance the Extensibility of Adaptive Tutoring Technologies*. in *The Interservice/Industry Training, Simulation & Education Conference (I/ITSEC)*. 2012. Orlando, FL: NTSA.

11. Merrill, M.D., *Component display theory*. Instructional-design theories and models: An overview of their current status, 1983. **1**: p. 282-333.
12. Rowe, J., et al. *Extending GIFT with a Reinforcement Learning-Based Framework for Generalized Tutorial Planning*. in *Generalized Intelligent Framework for Tutoring (GIFT) Users Symposium (GIFTSym4)*. 2016.
13. Chi, M.T. and R. Wylie, *The ICAP framework: Linking cognitive engagement to active learning outcomes.* Educational Psychologist, 2014. **49**(4): p. 219-243.
14. Patil, A.S. and A. Abraham, *Intelligent and Interactive Web-Based Tutoring System in Engineering Education: Reviews, Perspectives and Development*, in *Computational Intelligence for Technology Enhanced Learning. Studies in Computational Intelligence*, F. Xhafa, et al., Editors. 2010, Springer-Verlag.: Berlin. p. 79-97.
15. Aleven, V., et al., *Toward meta-cognitive tutoring: A model of help seeking with a Cognitive Tutor.* International Journal of Artificial Intelligence in Education, 2006. **16**(2): p. 101-128.
16. Graesser, A.C., et al., *AutoTutor.* Applied natural language processing: Identification, investigation, and resolution. Hershey, PA: IGI Global, 2012.
17. Sottilare, R.A., et al., *The Generalized Intelligent Framework for Tutoring (GIFT).* 2012.
18. Advanced Distributed Learning Initiative, *Sharable Content Object Reference Model (SCORM™).* Advanced Distributed Learning, http://www. adlnet. org, 2001.