The Assistment Builder: A Rapid Development Tool for ITS

Terrence E. TURNER, Michael A. MACASEK, Goss NUZZO-JONES, Neil T. HEFFERNAN
Worcester Polytechnic Institute
100 Institute Rd, Worcester, MA 01609
(508) 831-5569
tinylox@wpi.edu, macasek@wpi.edu, goss@wpi.edu, nth@wpi.edu
Ken KOEDINGER
Human-Computer Interaction Institute
Carnegie Mellon University, Pittsburgh, PA, USA

Abstract. Intelligent Tutoring Systems are notoriously costly to construct [1], and require PhD level experience in cognitive science and rule based programming. The goal of this research was to ease the development process for building pseudo-tutors [5], which are ITS constructs that mimic cognitive tutors but are limited in that they only work for a single problem. The Assistment Builder is a system designed to rapidly develop, test, and deploy simple pseudo-tutors. These tutors provide a simple cognitive model based upon a state graph tailored to a specific problem. These tutors offer many of the features of rule-based tutors, but without the expensive creation time. The system simplifies the process of tutor construction to allow users with little or no ITS experience to develop content. The system provides a web-based interface as a means to build and store these simple tutors we have called *Assistments*. This paper describes our attempt to make the process of developing content easy for teachers. We present some evidence to suggest that these novice users can develop a tutor for a problem in under thirty minutes.

1.0 Introduction

This research aims to develop tools for the rapid development and deployment of Intelligent Tutoring Systems (ITS). Specifically, this research focused on so-called "pseudo-tutors" that are a simplification of cognitive rule-based tutors [5]. Model tracing rule-based tutors [1] have been shown to be effective [6], but development time on them is highly prohibitive, from 100-1000 hours of development time per hour of content [7][1]. Development also requires a very specialized knowledge set. Tutor developers are required to be expert system programmers, in addition to developing the cognitive model, to say nothing of being a content expert. Another aim of this research was to make our tools accessible to novices, with no programming experience, and less than an hour of training.

A pseudo-tutor is a simplified cognitive model based on a state graph. Student actions trigger transitions in the graph, and the current state of the problem is stored by the graph. Pseudo-tutors have nearly identical behavior to a rule-based tutor, but suffer from having no ability to generalize to different problems [4]. This pseudo-tutor approach allows for predicted behaviors and provides feedback based on those behaviors. We also combined this state graph with a conceptually broader branching structure referred to as scaffolding. Scaffolding provides sub-problems to the initial question, often designed to address specific concepts within the initial question. This allows for a higher-level of predicted actions to be handled.

1.1 Purpose of the Assistment Builder

The Assistment Builder is an application supporting the Assistment Project [8]. We sought to create a tool that would provide a simple web-based interface for creating these pseudo-tutors that could rapidly be deployed across the web, and if errors were found with the tutor,

bug-fixing or correction would be quick and simple. The tool had to be usable by someone with no programming experience or ITS background. We wanted the teachers in the public school system to be able to build pseudo-tutors. These pseudo-tutors are often referred to as *Assistments*, but the term is not limited to pseudo-tutors.

A secondary purpose of the Assistment Builder was to aid the construction of a Transfer Model. A Transfer Model is a cognitive model construct divorced from specific tutors. The Transfer Model is a directed graph of *knowledge components* representing specific concepts that a student could learn. This allows us to maintain a complex cognitive model of the student without necessarily involving a production rule system.

When a user first begins to use the Assistment Builder they will be greeted by the standard blank skeleton question. The user can enter the question text, images, answers, and hint messages to complete the root question. After these steps the appropriate scaffolding is added. The question layout is separated into several views the *Main View*, *All Answer View*, *Correct Answer View*, *Incorrect Answer View*, *Hints View*, and *Transfer Model View*. Together these views allow a user to highly customize their question and its subsequent scaffolding.

2.0 Methods

To analyze the effectiveness of the Assistment Builder, we developed a system to log the actions of an author. Each action is recorded with associated meta-data, including author, timestamps, the specific series of problems being worked on, and data specific to each action. The authors were asked to build original items and keep track of roughly how much time spent on each item for corroboration. The authors were also asked to create "morphs," a term used to indicate a new problem that had a very similar setup to an existing problem. "Morphs" are usually constructed by loading the existing problem into the Assistment Builder, altering it, and saving it with a different name. This allows rapid content development for testing transfer between problems. We wanted to compare the development time for original items to that of "morphs" [8].

Another trial of the Assistment Builder with less rigorous methodology was testing how authors with little experience would react to the software. To test the usability of the Assistment Builder, we were able to provide the software to two high-school teachers in the Worcester, Massachusetts area. These teachers were computer literate, but had no previous experience with intelligent tutoring systems, or creating mathematics educational software. Our tutorial consisted of demonstrating the creation of a problem using the Assistment Builder, then allowing the teacher to create their own with an experienced observer to answer questions.

3.0 Results & Analysis

Prior to the implementation of logging within the Assistment Builder, we obtained encouraging anecdotal results of the software's use. A high-school mathematics teacher was able to create 15 items and morph each one, resulting in 30 *Assistments* over several months. Her training consisted of approximately four hours spread over two days in which she created 5 original *Assistments* under supervision. While there is unfortunately no log data to strengthen this result, it is nonetheless encouraging.

The logging data obtained suggests that the average time to build an entirely new *Assistment* is approximately 25 minutes. This data was acquired by examining the time that elapsed between the initialization of a new problem and the problem save time. Creation times for *Assistments* with more scaffolds naturally took longer than those with fewer scaffolds. Experience with the system also decreases *Assistment* creation time, as end-users who are more comfortable with the Assistment Builder are able work faster. Nonetheless, even users who were just learning the system were able to create *Assistments* in reasonable

time. For instance, Users 2, 3, and 4 (see Table 1) provide examples of end-users who have little experience using the Assistment Builder. In fact, some of them are using the system for the first time in the examples provided.

Username	Number of Scaffolds	Time Elapsed (min)
User 1	10	35
User 1	2	23
User 2	3	45
User 2	2	31
User 2	0	8
User 3	2	21
User 4	3	37
User 4	0	15
User 5	4	30
User 5	2	8
User 5	4	13
User 5	4	35
User 5	3	31
User 5	2	24
		Average: 25.4 minutes

Table 1 - Full Item Creation

We were also able to collect useful data on morph creation time and Assistment editing time. On average morphing an *Assistment* takes approximately 10-20 minutes depending on the number of scaffolds in an Assistment and the nature of the morph.

4.0 Conclusions

The Assistment Builder has been in use over six months by a variety of users involved in the Assistments project. Teachers, developers, and others have used it to develop pseudo-tutor *Assistments*. The end result has been over a thousand individual pseudo-tutors deployed on the web. The breadth of users who developed these *Assistments* and the number created would not have been possible without the Assistment Builder.

References

1. Anderson, J. R. (1993). Rules of the mind. Hillsdale, NJ: Erlbaum.
2. 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.
3. Jackson, G.T., Person, N.K., and Graesser, A.C. (2004) Adaptive Tutorial Dialogue in AutoTutor. *Proceedings of the workshop on Dialog-based Intelligent Tutoring Systems at the 7th International conference on Intelligent Tutoring Systems. Universidade Federal de Alagoas, Brazil, 9-13*.
4. Jarvis, M., Nuzzo-Jones, G. & Heffernan. N. T. (2004) Applying Machine Learning Techniques to Rule Generation in Intelligent Tutoring Systems. Proceedings of 7th Annual Intelligent Tutoring Systems Conference, Maceio, Brazil. Pages 541-553
5. Koedinger, K. R., Aleven, V., Heffernan. T., McLaren, B. & Hockenberry, M. (2004) Opening the Door to Non-Programmers: Authoring Intelligent Tutor Behavior by Demonstration. *Proceedings of 7th Annual Intelligent Tutoring Systems Conference, Maceio, Brazil*. Page 162-173
6. Koedinger, K. R., Anderson, J. R., Hadley, W. H., & Mark, M. A. (1997). Intelligent tutoring goes to school in the big city. *International Journal of Artificial Intelligence in Education*, *8,* 30-43.
7. Murray, T. (1999). Authoring intelligent tutoring systems: An analysis of the state of the art. International Journal of Artificial Intelligence in Education, 10, pp. 98-129.
8. Razzaq, L., Feng, M., Nuzzo-Jones, G., Heffernan, N.T., Aniszczyk, C., Choksey, S., Livak, T., Mercado, E., Turner, T.E., Upalekar. R, Walonoski, J.A., Macasek. M.A., Rasmussen, K.P. (2005) The Assistment Project: Blending Assessment and Assisting. *Submitted to the 12th Annual Conference on Artificial Intelligence in Education 2005, Amsterdam.*