

## A Web-based Authoring Tool for Intelligent Tutors: Blending Assessment and Instructional Assistance

Leena Razzaq<sup>1</sup>, Mingyu Feng<sup>1</sup>, Neil T. Heffernan<sup>1</sup>, Kenneth R. Koedinger<sup>2</sup>, Brian Junker<sup>2</sup>, Goss Nuzzo-Jones<sup>1</sup>, Michael A. Macasek<sup>1</sup>, Kai P. Rasmussen<sup>1</sup>, Terrence E. Turner<sup>1</sup>, and Jason A. Walonoski<sup>1</sup>

<sup>1</sup> Worcester Polytechnic Institute, 100 Institute Road, Worcester, Massachusetts, USA

<sup>2</sup> Carnegie Mellon University, 5000 Forbes Ave., Pittsburgh, Pennsylvania, USA  
assistments@wpi.edu

Middle school mathematics teachers are often forced to choose between assisting students' development and assessing students' abilities because of limited classroom time available. To help teachers make better use of their time, a web-based system, called the Assistment system, was created to integrate assistance and assessment by offering instruction to students while providing a more detailed evaluation of their abilities to the teacher than is possible under current approaches. An initial version of the Assistment system was created and used in May, 2004 with approximately 200 students and over 1000 students currently use it once every two weeks. The hypothesis is that Assistments can assist students while also assessing them. This chapter describes the Assistment system and some preliminary results.

### 2.1 Introduction

Limited classroom time available in middle school mathematics classes compels teachers to choose between time spent assisting students' development and time spent assessing students' abilities. To help resolve this dilemma, assistance and assessment are integrated in a web-based system called the Assistment<sup>3</sup> System that will offer instruction to students while providing a more detailed evaluation of their abilities to the teacher than is possible under current approaches. The plan is for students to work on the Assistment website for about 20 minutes per week. As building intelligent tutoring systems can be

---

<sup>3</sup> The term Assistment was coined by Kenneth Koedinger and blends Assisting and Assessment.

very costly [15], the Office of Naval Research provided funding to reduce those costs. We reported on the substantial reductions in time needed to build intelligent tutoring systems with the tools we have built.<sup>4</sup> The Assistment system is an artificial intelligence program and each week when students work on the website, the system “learns” more about the students’ abilities and thus, it can hypothetically provide increasingly accurate predictions of how they will do on a standardized mathematics test. The Assistment System is being built to identify the difficulties individual students - and the class as a whole - are having. It is intended that teachers will be able to use this detailed feedback to tailor their instruction to focus on the particular difficulties identified by the system. Unlike other assessment systems, the Assistment technology also provides students with intelligent tutoring assistance while the assessment information is being collected.

An initial version of the Assistment system was created and tested in May, 2004. That version of the system included 40 Assistment items. There are now over 700 Assistment items. The key feature of Assistments is that they provide instructional assistance in the process of assessing students. The hypothesis is that Assistments can do a better job of assessing student knowledge limitations than practice tests or other on-line testing approaches by using a “dynamic assessment” approach. In particular, Assistments use the amount and nature of the assistance that students receive as a way to judge the extent of student knowledge limitations.

The rest of this chapter covers 1) the web-based architecture we used that students and teachers interact with, 2) the Builder application that we use internally to create this content and finally 3) a report on the designing of the content and the evaluation of the assistance and assessment that the Assistment system provides.

## 2.2 The Extensible Tutor Architecture

The eXtensible Tutor Architecture (XTA) is a framework that controls the interface and behaviors of our intelligent tutoring system via a collection of modular units. These units conceptually consist of a curriculum unit, a problem unit, a strategy unit, and a logging unit. Each conceptual unit has an abstract and extensible implementation allowing for evolving tutor types and content delivery methods. The XTA is represented by the diagram given in Figure 1, illustrating the actual composition of the units. This diagram shows

<sup>4</sup> This research was made possible by the US Dept of Education, Institute of Education Science, “Effective Mathematics Education Research” program Grant No. R305K03140, the Office of Naval Research Grant No. N00014-03-1-0221, NSF CAREER award to Neil Heffernan, and the Spencer Foundation. Author Razzaq was funded by the National Science Foundation under Grant No. 0231773. All the opinions in this article are those of the authors, and not those of any of the funders.

the relationships between the different units and their hierarchy. Within each unit, the XTA has been designed to be highly flexible in anticipation of future tutoring methods and interface layers. This was accomplished through encapsulation, abstraction, and clearly defined responsibilities for each component. These software engineering practices allowed us to present a clear developmental path for future components. That being said, the current implementation has full functionality in a variety of useful contexts.

### 2.2.1 Curriculum Unit

The curriculum unit can be conceptually subdivided into two main pieces: the curriculum itself, and sections. The curriculum is composed of one or more sections, with each section containing problems or other sections. This recursive structure allows for a rich hierarchy of different types of sections and problems.

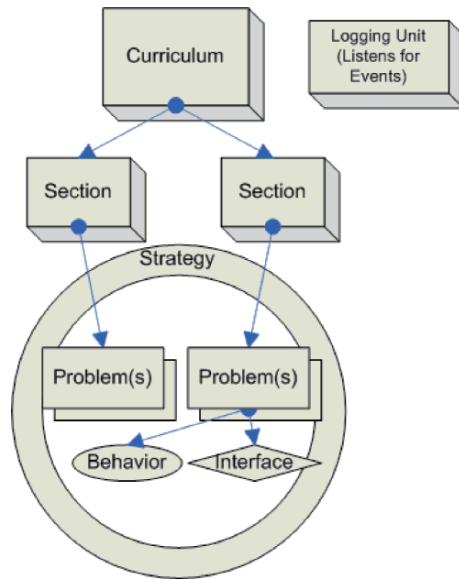
Progress within a particular curriculum, and the sections of which it is composed, are stored in a progress file - an XML meta-data store that indexes into the curriculum and the current problem (one progress file per student per curriculum).

The section component is an abstraction for a particular listing of problems. This abstraction has been extended to implement our current section types, and allows for future expansion of the curriculum unit. Currently existing section types include “Linear” (problems or sub-sections are presented in linear order), “Random” (problems or sub-sections are presented in a pseudo-random order), and “Experiment” (a single problem or sub-section is selected pseudo-randomly from a list, the others are ignored). Plans for future section types include a “Directed” section, where problem selection is directed by the student’s knowledge model [2].

### 2.2.2 Problem Unit

The problem unit represents a problem to be tutored, including questions, answers, and relevant knowledge-components required to solve the problem. For instance pseudo-tutors are a hierarchy of questions connected by correct and incorrect answers, along with hint messages and other feedback. Each of these questions are represented by a problem composed of two main pieces: an interface and a behavior.

The interface definition is interpreted by the runtime and displayed for viewing and interaction to the user. This display follows a two-step process, allowing for easy customization of platform and interface specification. The interface definition consists of high-level interface elements (“widgets”), which can have complex behavior (multimedia, spell-checking text fields, algebra parsing text fields). These “high-level” widgets have a representation in the runtime composed of “low-level” widgets. “Low-level” widgets are widgets common to many possible platforms of interface, and include text labels, text



**Fig. 2.1.** Abstract unit diagram

fields, images, radio buttons, etc. These “low-level” widgets are then consumed by an interface display application. Such applications consume “low-level” widget XML, and produce an interface on a specific platform. The event model (described below) and relationship of “high-level” to “low-level” widgets allow a significant degree of interface customizability even with the limitations of HTML. Other technologies, such as JavaScript and streaming video are presently being used to supplement our interface standard. Future interface display applications are under consideration, such as Unreal Tournament for Warrior Tutoring [12], and Macromedia Flash for rich content definition.

The behaviors for each problem define the results of actions on the interface. An action might consist of pushing a button or selecting a radio button. Examples of behavior definitions are state graphs, cognitive model tracing, or constraint tutoring, defining the interaction that a specific interface definition possesses. To date, state graph or pseudotutor definitions have been implemented in a simple XML schema, allowing for a rapid development of pseudo tutors [16]. We have also implemented an interface to the JESS (Java Expert System Shell) production system, allowing for full cognitive model behaviors. A sample of the type of cognitive models we would wish to support is outlined in Jarvis et al [9]. The abstraction of behaviors allows for easy extension of both their functionality and by association their underlying XML definition.

Upon user interaction, a two-tiered event model (see Figure 2) is used to respond to that interaction. These tiers correspond to the two levels of

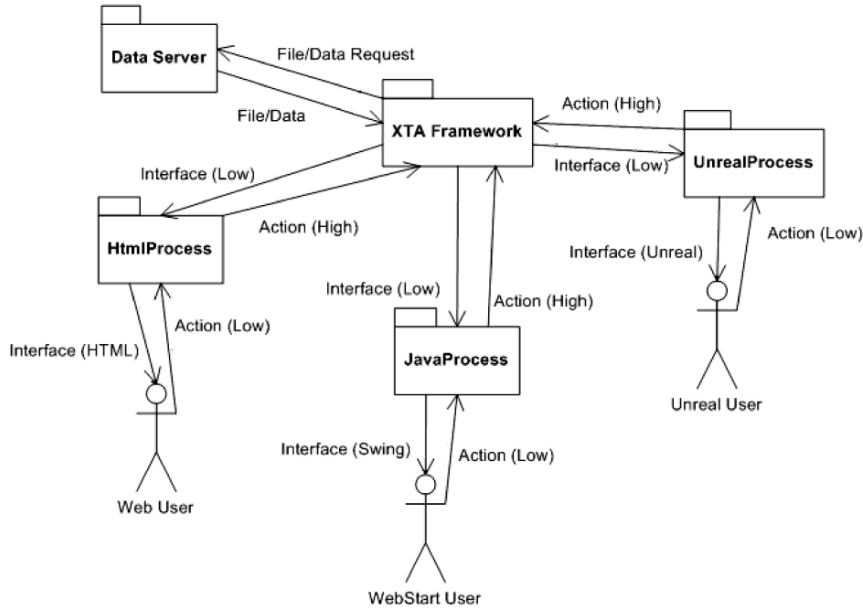
widgets described above, and thus there are “high-level” actions and “low-level” actions. When the user creates an event in the interface, it is encoded as a “low-level” action and passed to the “high-level” interface widget. The “high-level” interface widget may (or may not) decide that the “low-level” action is valid, and encode it as a “high-level” action. An example of this is comparing an algebra text field (scripted with algebraic equality rules) with a normal text field by initiating two “low-level” actions such as entering “3+3” and “6” in each one. The algebra text field would consider these to be the same “high-level” action, whereas a generic text field would consider them to be different “high-level” actions. “High-level” actions are processed by the interpreted behavior and the interface is updated depending on the behavior’s response to that action. The advantage of “high-level” actions is that they allow an interface widget or content developer to think in actions relevant to the widget, and avoid dealing with a large number of trivial events.

### 2.2.3 Strategy Unit

The strategy unit allows for high-level control over problems and provides flow control between problems. The strategy unit consists of tutor strategies and the agenda. Different tutor strategies can make a single problem behave in different fashions. For instance, a scaffolding tutor strategy arranges a number of problems in a tree structure, or scaffold. When the student answers the root problem incorrectly, a sequence of other problems associated with that incorrect answer is queued for presentation to the student. These scaffolding problems can continue to branch as the roots of their own tree. It is important to note that each problem is itself a self-contained behavior, and may be an entire state graph/pseudo-tutor, or a full cognitive tutor.

Other types of tutor strategies already developed include message strategies, explain strategies, and forced scaffolding strategies. The message strategy displays a sequence of messages, such as hints or other feedback or instruction. The explain strategy displays an explanation of the problem, rather than the problem itself. This type of tutoring strategy would be used when it was already assumed that the student knew how to solve the problem. The forced scaffolding strategy forces the student into a particular scaffolding branch, displaying but skipping over the root problem. The concept of a tutor strategy is implemented in an abstract fashion, to allow for easy extension of the implementation in the future. Such future tutor strategies could include dynamic behavior based on knowledge tracing of the student log data. This would allow for continually evolving content selection, without a predetermined sequence of problems.

This dynamic content selection is enabled by the agenda. The agenda is a collection of problems arranged in a tree, which have been completed or have been queued up for presentation. The contents of the agenda are operated upon by the various tutor strategies, selecting new problems from sections



**Fig. 2.2.** Network architecture and event model diagram.

(possibly within sections) within a curriculum to append and choosing the next problem to travel to [7].

#### 2.2.4 Logging Unit

The final conceptual unit of the XTA is the logging unit with full-featured relational database connectivity. The benefits of logging in the domain of ITS have been acknowledged, significantly easing data mining efforts, analysis, and reporting [14]. Additionally, judicious logging can record the data required to replay or rerun a user's session.

The logging unit receives detailed information from all of the other units relating to user actions and component interactions. These messages include notification of events such as starting a new curriculum, starting a new problem, a student answering a question, evaluation of the student's answer, and many other user-level and framework-level events.

Capturing these events has given us an assortment of data to analyze for a variety of needs. User action data captured allows us to examine usage-patterns, including detection of system gaming (superficially going through tutoring content without actually trying to learn) [7]. This data also enables us to quickly build reports for teachers on their students, as well as giving a complete trace of student work. This trace allows us to replay a user's session, which could be useful for quickly spotting fundamental misunderstandings on

the part of the user, as well as debugging the content and the system itself (by attempting to duplicate errors).

The logging unit components are appropriately networked to leverage the benefits of distributing our framework over a network and across machines. The obvious advantage this provides is scalability.

### 2.2.5 System Architecture

The XTA provides a number of levels of scalability. To allow for performance scalability, care was taken to ensure a low memory footprint. It is anticipated, based on simple unit testing, that thousands of copies of the XTA could run on a single machine. More importantly, the individual units described above are separated by network connections (see Figure 2). This allows individual portions of the XTA to be deployed on different computers. Thus, in a server context, additional capacity can be added without software modification, and scalability is assured.

The runtime can also transform with little modification into a client application or a server application instantiated over a web server or other network software launch, such as Java WebStart. Both types of applications allow for pluggable client interfaces due to a simple interface and event model, as described in the interface unit. A client side application contains all the network components described above (Figure 2) as well as content files required for tutoring, and has the capacity to contact a remote logging unit to record student actions. Running the XTA in a server situation results in a thin client for the user (at present either HTML or Java WebStart), which operates with the interface and event model of the server. Thus the server will run an instance of the XTA for every concurrent user, illustrating the need for a small memory footprint. The XTA instances on the server contact a centralized logging unit and thus allow for generated reports available through a similar server [4].

### 2.2.6 Methods

The XTA has been deployed as the foundation of the Assistments Project [12]. This project provides mathematics tutors to Massachusetts students over the web and provides useful reports to teachers based on student performance and learning. The system has been in use for three years, and has had thousands of users. These users have resulted in over 1.3 million actions for analysis and student reports [4]. To date, we have had a live concurrency of approximately 50 users from Massachusetts schools. However, during load testing, the system was able to serve over 500 simulated clients from a single J2EE/database server combination. The primary server used in this test was a Pentium 4 with 1 gigabyte of RAM running Gentoo Linux. Our objective is to support 100,000 students across the state of Massachusetts. 100,000 students divided across 5 school days would be 20,000 users a day. Massachusetts schools generally have 7 class periods, which would be roughly equivalent to supporting 3,000 users

concurrently. This calculation is clearly based on estimations, and it should be noted that we have not load tested to this degree.

Tutors that have been deployed include scaffolding state diagram pseudo-tutors with a variety of strategies (see Figure 3 for a pseudo-tutor in progress). We have also deployed a small number of JESS cognitive tutors for specialized applications. It should be noted that the tutors used in the scaling test described above were all pseudo-tutors, and it is estimated that a much smaller number of JESS tutors could be supported.

In summary, the launch of the XTA has been successful. The configuration being used in the Assistments project is a central server as described above, where each student uses a thin HTML client and data is logged centrally. The software has been considered stable for several months, and has been enthusiastically reviewed by public school staff. Since September 2004, the software has been in use at least three days a week over the web by a number of schools across central Massachusetts. This deployment is encouraging, as it demonstrates the stability and initial scalability of the XTA, and provides significant room to grow.

Use the figure above to answer the questions.

What is the measure of angle A?

Hmm, no.  
Let me break this down for you.  
First you need to find the measure of angle BCA. What do you think it is?

Angles BCD and BCA are supplementary. That means their sum is 180 degrees.

Fig. 2.3. Pseudo-tutor in progress.

The larger objective of this research was to build a framework that could support 100,000 students using ITS software across the state of Massachusetts. We're encouraged by our initial results from the Assistments Project, which indicate that the XTA has graduated from conceptual framework into a usable platform (available at <http://www.assistments.org>). However, this test of the software was primarily limited to pseudo-tutors, though model-tracing tutors are supported. One of the significant drawbacks of model-tracing tutors in a

server context is the large amount of resources they consume. This resource consumption would prohibit scaling to the degree that is described in our results. A partial solution to this might be the support of constraint-based tutors [10], which could conceivably take fewer resources, and we are presently exploring this concept. These constraint tutors could take the form of a simple JESS model (not requiring an expensive model trace), or another type of scripting language embedded in the state-graph pseudo-tutors.

Other planned improvements to the system include dynamic curriculum sections, which will select the next problem based on the student's performance (calculated from logged information). Similarly, new tutor strategies could alter their behavior based on knowledge tracing of the student log data. Also, new interface display applications are under consideration, using the interface module API. As mentioned, such interfaces could include Unreal Tournament<sup>TM</sup>, Macromedia Flash<sup>TM</sup>, or a Microsoft .NET application. We believe the customizable nature of the XTA could make it a valuable tool in the continued evolution of Intelligent Tutoring Systems.

## 2.3 The Assistment Builder

The foundation of the Assistment architecture is the content representation, an XML (eXensible Markup Language) schema that defines each problem. A problem consists of an interface definition and behavior definition. The interface definition provides a collection of simple widgets to be displayed to the student. The behavior definition is a representation of the state graph and its transitions, or a cognitive model (e.g. JESS rules). Many types of behaviors are possible within the representation and architecture. These two parts of the representation are consumed by the runtime Assistment architecture, and presented to the student over the web. Student actions are then fed back to the representation, and compared with the state graph or used to model trace.

### 2.3.1 Purpose of the Assistment Builder

The XML representation of content provides a base for which we can rapidly create specific pseudo-tutors. We sought to create a tool that would provide a simple web-based interface for creating these pseudo-tutors. Upon content creation, we could rapidly deploy the tutor across the web, and if errors were found with the tutor, bug-fixing or correction would be quick and simple. Finally, the tool had to be usable by someone with no programming experience and no ITS background. This applied directly to our project of creating tutors for the mathematics section of the Massachusetts Comprehensive Assessment System (MCAS) test [10]. 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. These knowledge components are then associated with a specific tutor (or even sub-question within that tutor) so that the tutor is associated with a number of knowledge components. This allows us to maintain a complex cognitive model of the student without necessarily involving a production rule system. It also allows analysis of student performance in the context of the Transfer Model, resulting in knowledge tracing [2] and other useful methods. The simplest way to “mark” tutors in a Transfer Model is to associate the tutors (or their sub-questions) with knowledge components when the tutors are created. The Transfer Model created by the Assistment team is used to classify 8th grade mathematics items and has approximately 90 knowledge components. Over the six months since the inception of the Assistment Builder, nearly 1000 individual problems have thus far been associated with these 90 knowledge components.

### 2.3.2 Assistments

The basic structure of an Assistment is a top-level question that can then branch to scaffolding questions based on student input. The Assistment Builder interface uses only a subset of the full content XML representation, with the goal of producing simple pseudo-tutors. Instead of allowing arbitrary construction of question interfaces there are only five widget choices available to a content creator. These are radio-buttons, pull-down menus, checkboxes, text-fields, and algebra text fields that automatically evaluate mathematical expressions. The state graphs for each question are limited to two possible states. An arc occurs between the two states when the end-user answers a question properly. The student will remain in the initial state until the question is answered properly or skipped programmatically.

The scaffolding questions mentioned above are all queued as soon as a user gets the top-level question incorrect, or requests help in the form of a hint (for either event, the top-level question is skipped). Upon successfully completing the displayed scaffolding question the next is displayed until the queue is empty. Once the queue is empty, the problem is considered complete. This type of linear Assistment can be easily made with our tool, by first creating the main item and then the subsequent scaffolding questions. When building an Assistment a user may also add questions that will appear when a specific incorrect answer is received. This allows branches to be built that tutor along a “line of reasoning” in a problem, which adds more generality than a simple linear Assistment. Many Assistment authors also use text feedback on certain incorrect answers. These feedback messages are called buggy messages. Buggy messages address the specific error made, and match common or expected mistakes.

Content creators can also use the Assistment Builder to add hint messages to problems, providing the student with hints attached to a specific scaffolding question. This combination of hints, buggy messages, and branched scaffolding questions allow even the simple state diagrams described above to assume a useful complexity. Assistments constructed with the Assistment Builder can provide a tree of scaffolding questions branched from a main question. Each question consists of a customized interface, hint messages and bug messages, along with possible further branches.

### 2.3.3 Web Deployment

We constructed the Assistment Builder as a web application for accessibility and ease of use. A teacher or content creator can create, test, and deploy an Assistment without installing any additional software. Users can design and test their Assistments and then instantly deploy them. If further changes or editing are needed the Assistment can be loaded into the builder, modified, and then redeployed across all the curriculums that make use of the tutor. By making the Assistment Builder available over the web, there is no need for users to update any software if a new feature is added. They reap the benefits of any changes made to the system as soon as they log on. By storing created Assistments locally on our servers, allowing end-users to easily create a curriculum and assign it to a class for use by students is a simple task.

### 2.3.4 Features

Though there are many significant improvements to be made to the Assistment Builder's user interface, it is usable and reasonably easy to learn. When users first begin to use the Assistment Builder they will be greeted by the standard blank skeleton question. The initial blank skeleton question will be used to create the root question. The user will 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 users to highly customize their questions and their subsequent scaffolding.

Initially the user is presented with the Main View (see Figure 4). In this view question text, correct answers, and images can be added to the question. Additionally the user can add new scaffolding off of the current question, and specify if they would like the answers to be in a sorted or random order. The Main View is designed to gather the absolute minimum information needed to generate a question.

Upon completion of the items in the Main View the user then has the option to move to other views in order to further refine the question. Typically the next view to complete is the All Answer View. In the All Answers View

the user has the option to add additional correct answers as well as incorrect answers. The incorrect answers serve two purposes. First, they allow a teacher to specify the answers students are likely to choose incorrectly and provide feedback in the form of a message or scaffolding. Second, the user can populate a list of answers for multiple choice questions. The user now has the

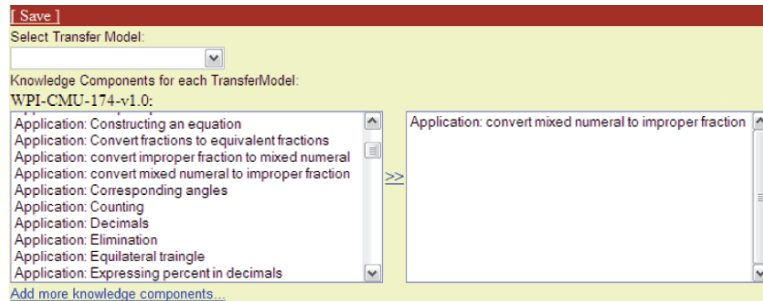
The screenshot displays the Assistent builder interface in three stages:

- Initial Question View:** Shows a main question editor with text "what is 3/4 of 1 1/2?". It includes a "Correct Answer" field with "1 1/8:" and a "Customize this Question" menu with options like "Edit hint messages", "Edit expected incorrect answers", "Edit correct answers", "Edit all answers", and "Edit Transfer Model". Below the question are options for "End line of questioning", "Ask next question in line of questioning", and "Delete this question and all of its scaffolds".
- Scaffold View:** Shows a scaffolded question: "In the statement above, what mathematical operation does the word *of* represent?". The "Correct Answer" field contains "Multiplication:". The same customization and scaffolding options are present.
- Incorrect Answer View:** A "Save" button is highlighted. It shows options for handling incorrect answers: "Comment on wrong answer" (with a "Some Default Message" field), "Question on wrong answer", and "Add more expected incorrect answers...".

**Fig. 2.4.** The Assistent builder - initial question, one scaffold and incorrect answer view.

option to specify a message to be displayed for an incorrect answer or the option to scaffold. If the scaffolding option is chosen a new question block will appear indented below the current question. In the Hints View messages can be created that will be presented to the student when a hint is requested. Hints can consist of text, an image, or both. Multiple hint messages can be entered; one message will appear per hint request in the order that they are listed in this view. The final view is the Transfer Model View (see Figure 5). In this view the user has the option of specifying one or more skills that are associated with this particular question.

As mentioned above there are two methods of providing scaffolding questions: either by selecting Ask Next Line of Questioning from the Main View



**Fig. 2.5.** Transfer model view.

or specify scaffolding on a specific incorrect answer. In utilizing either of these methods a new skeleton question will be inserted into the correct position below the current question. Creating a scaffolding question is done in the exact manner as the root question. After saving the Assistment the tutor is ready to be used. An Assistment can be modified at any time by loading it into the Assistment Builder and changing its properties accordingly. A completed running Assistment can be seen in Figure 6.

### 2.3.5 Methods

To analyze the effectiveness of the Assistment Builder, we developed a system to log the actions of an author. While authors have been constructing items for nearly six months, only very recently has the Assistment Builder had the capability to log actions.

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. These actions were recorded for a number of Assistment authors over several days. 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” [10]. 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 teachers to create their own with an

What is  $\frac{3}{4}$  of  $1\frac{1}{2}$ ?

Hmm, no.

Let me break this down for you.

In the statement above, what mathematical operation does the word *of* represent?

Subtraction  Division  Multiplication  Addition

Express  $1\frac{1}{2}$  as an improper fraction:

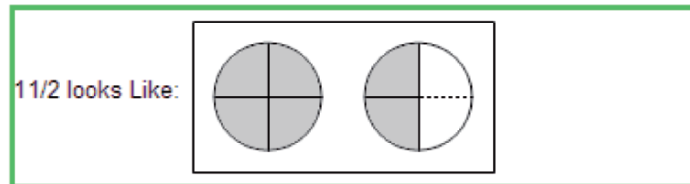


Fig. 2.6. An Assistment running.

experienced observer to answer questions. Finally, we allowed them to author Assistments on their own, without assistance.

### 2.3.6 Results and 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. Entirely new Assistments are those that are built using new content and not based on existing material. This data was acquired by examining the time that elapsed between the initialization of a new problem and when it was saved. 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 to 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.

**Table 2.1.** Full Item Creation

Username	Number of scaffolds	Time elapsed (min)
User1	10	35
User1	2	23
User2	3	45
User2	2	31
User2	0	8
User3	2	21
User4	3	37
User4	0	15
User5	4	30
User5	2	8
User5	4	13
User5	4	35
User5	3	31
User5	2	24
		Average: 25.4 minutes

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. More complex Assistment morphs require more time because larger parts of an Assistment must be changed. Editing tasks usually involve minor changes to an Assistment's wording or interface. These usually take less than a minute to locate and fix.

### 2.3.7 Future Work

In our continuing efforts to provide a tool that is accessible to even the most novice users we are currently working on two significant enhancements to the Assistment Builder. The first enhancement is a simplified interface that is both user-friendly and still provides the means to create powerful scaffolding pseudo-tutors. The most significant change to the current interface is the addition of a tab system that will allow the user to clearly navigate the different components of a question. The use of tabs allows us to present the user with

only the information related to the current view, reducing the confusion that sometimes takes place in the current interface.

The second significant enhancement is a new question type. This question type will allow a user to create a question with multiple inputs of varying type. The user will also be able to include images and Macromedia Flash movies. Aside from allowing multiple answers in a single question, the new question type allows a much more customizable interface for the question. Users can add, in any order, a text component, a media component, or an answer component. The ability to place a component in any position in the question will allow for a more “fill in the blank” feel for the question and provide a more natural layout. This new flexibility will no longer force questions into the text, image, answer format that is currently used.

## 2.4 Content Development and Usage

In December of 2003, we met with the Superintendent of the Worcester Public Schools in Massachusetts, and were subsequently introduced to the three math department heads of 3 out of 4 Worcester middle schools. The goal was to get these teachers involved in the design process of the Assistment System at an early stage. The main activity done with these teachers was meeting about one hour a week to do “knowledge elicitation” interviews, whereby the teachers helped design the pedagogical content of the Assistment System.

### 2.4.1 Content Development

The procedure for knowledge elicitation interviews went as follows. A teacher was shown a Massachusetts Comprehensive Assessment System (MCAS) test item and asked how she would tutor a student in solving the problem. What kinds of questions would she ask the student? What hints would she give? What kinds of errors did she expect and what would she say when a student made an expected error? These interviews were videotaped and the interviewer took the videotape and filled out an “Assistment design form” from the knowledge gleaned from the teacher. The Assistment was then implemented using the design form. The first draft of the Assistment was shown to the teacher to get her opinion and she was asked to edit it. Review sessions with the teachers were also videotaped and the design form revised as needed. When the teacher was satisfied, the Assistment was released for use by students. For instance, a teacher was shown a MCAS item on which her students did poorly, such as item number 19 from the year 2003, which is shown in Figure 7. About 15 hours of knowledge elicitation interviews were used to help guide the design of Assistments.

Figure 8 shows an Assistment that was built for item 19 of 2003 shown above. Each Assistment consists of an original item and a list of scaffolding questions (in this case, 5 scaffolding questions). The first scaffolding question

- 19 Triangles  $ABC$  and  $DEF$  shown below are congruent.



The perimeter of  $\triangle ABC$  is 23 inches. What is the length of side  $\overline{DF}$  in  $\triangle DEF$ ?

Fig. 2.7. Item 19 from the 2003 MCAS.

appears only if the student gets the item wrong. Figure 8 shows that the student typed “23” (which happened to be the most common wrong answer for this item from the data collected). After an error, students are not allowed to try the item further, but instead must then answer a sequence of scaffolding questions (or “scaffolds”) presented one at a time.<sup>5</sup> Students work through the scaffolding questions, possibly with hints, until they eventually get the problem correct. If the student presses the hint button while on the first scaffold, the first hint is displayed, which would have been the definition of congruence in this example. If the student hits the hint button again, the hint describes how to apply congruence to this problem. If the student asks for another hint, the answer is given. Once the student gets the first scaffolding question correct (by typing AC), the second scaffolding question appears.

If the student selected  $1/2 * 8x$  in the second scaffolding question, a buggy message would appear suggesting that it is not necessary to calculate area. (Hints appear on demand, while buggy messages are responses to a particular student error). Once the student gets the second question correct, the third appears, and so on. Figure 8 shows the state of the interface when the student is done with the problem as well as a buggy message and two hints for the 4th scaffolding question.

About 200 students used the system in May 2004 in three different schools from about 13 different classrooms. The average length of time was one class period per student. The teachers seemed to think highly of the system and, in particular, liked that real MCAS items were used and that students received instructional assistance in the form of scaffolding questions. Teachers also like that they can get online reports on students’ progress from the Assistent web site and can even do so while students are using the Assistent System in their classrooms. The system has separate reports to answer the following questions about items, student, skills and student actions: Which items are my students

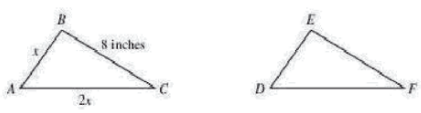
<sup>5</sup> As future work, once a predictive model has been built and is able to reliably detect students trying to “game the system” (e.g., just clicking on answer) students may be allowed to re-try a question if they do not seem to be “gaming”. Thus, studious students may be given more flexibility.

Welcome to Assisments Project Web Portal - Microsoft Internet Explorer

Assisments

Home

Triangles  $ABC$  and  $DEF$  shown below are congruent.



Triangles  $ABC$  and  $DEF$  are congruent.  
The perimeter of triangle  $ABC$  is 23 inches. What is the length of side  $DF$  in triangle  $DEF$ ?

Hmm, no.  
Let me break this down for you.

---

Which side of triangle  $ABC$  has the same length as side  $DF$  of triangle  $DEF$ ?

---

What is the perimeter of triangle  $ABC$ ?

$2x + 8$   
  $2x + x + 8$   
  $1/2 * 8x$   
  $1/2 * x(2x)$

---

Now, given the perimeter of triangle  $ABC$  equals 23 inches, you can write the equation  $2x + x + 8 = 23$  and solve it for  $x$ .   
What is the value of  $x$ ?

---

Remember, we are looking for side  $DF$ . Enter the length of side  $DF$ :

**$x=5$  is true but not what you are asked. Yes, side  $AB$  is 5 but what is  $DF$ ?**

The figure tells us that the length of side  $AC$  is  $2x$ . Knowing that  $x = 5$ , what is  $2x$ ?  
The answer is 10.

**Fig. 2.8.** An Assisment show just before the student hits the “done” button, showing two different hints and one buggy message that can occur at different points.

finding difficult? Which items are my students doing worse on compared to the state average? Which students are 1) doing the best, 2) spending the most time, 3) asking for the most hints etc.? Which of the approximately 90 skills that we are tracking are students doing the best/worst on? What are the exact actions that a given student took?

The three teachers from this first use of the Assistment System were impressed enough to request that all the teachers in their schools be able to use the system the following year. Currently that means that about 1,000 students are using the system for about 20 minutes per week for the 2004-2005 school year. Two schools have been using the Assistment System since September. A key feature of the strategy for both teacher recruitment and training is to get teachers involved early in helping design Assistments through knowledge elicitation and feedback on items that are used by their students.

We have spent considerable time observing its use in classrooms; for instance, one of the authors has logged over 50 days, and was present at over 300 classroom periods. This time is used to work with teachers to try to improve content and to work with students to note any misunderstandings they sometimes bring to the items. For instance, if it is noted that several students are making similar errors that were not anticipated, the Assistment Builder can be logged into and a buggy message added that addresses the students' misconception.

#### **2.4.2 Database Reporting**

The Assistment system produces reports individually for each teacher. These reports can inform the teacher about 1) "Which of the 90 skills being tracked are the hardest? 2) Which of the problems are students doing the poorest at and 3) reports about individual students. Figure 9 shows the "Grade book" report that shows for each student the amount of time spent in the system, the number of items they did, and their total score. Teachers can click on refresh and get instant updates. One of the common uses of this report is to track how many hints each student is asking for. We see that "Mary" has received a total of 700 over the course of 4 hours using the system, which suggests to teachers Mary might be using the system's help too much, but at this point it is hard to tell, given that Mary is doing poorly already.

#### **2.4.3 Analysis of data to determine whether the system reliably predicts MCAS performance**

One objective the project had was to analyze data to determine whether and how the Assistment System can predict students' MCAS performance. In Bryant, Brown and Campione [2], they compared traditional testing paradigms against a dynamic testing paradigm. In the dynamic testing paradigm a student would be presented with an item and when the student appeared to not be making progress, would be given a prewritten hint. If the

Student Name	Elapsed time (hh:mm)	Original Items				Scaffolding + Original Items		
		# Done	% Correct	MCAS Score	Performance Level	# Done	% Correct	# Hint Req.
Tom*	4:12	90	38%	214	Warning/Failing-High	228	44%	233
Dick*	4:01	98	66%	244	Pro./Adv.	158	59%	58
Harry*	4:07	58	40%	219	Needs improv.-Low	154	38%	77
Mary*	4:17	114	20%	200	Warning/Failing-Low	356	20%	705
Jack*	3:53	104	41%	214	Warning/Failing-High	267	43%	227

Fig. 2.9. The grade book report.

student was still not making progress, another prewritten hint was presented and the process was repeated. In this study they wanted to predict learning gains between pretest and posttest. They found that static testing was not as well correlated ( $R = 0.45$ ) as with their “dynamic testing” ( $R = 0.60$ ).

Given the short use of the system in May, 2004, there was an opportunity to make a first pass at collecting such data. The goal was to evaluate how well on-line use of the Assistment System, in this case for only about 45 minutes, could predict students’ scores on a 10-item post-test of selected MCAS items. There were 39 students who had taken the post-test. The paper and pencil post-test correlated the most with MCAS scores with an R-value of 0.75.

A number of different metrics were compared for measuring student knowledge during Assistment use. The key contrast of interest is between a static metric that mimics paper practice tests by scoring students as either correct or incorrect on each item, with a dynamic assessment metric that measures the amount of assistance students need before they get an item correct. MCAS scores for 64 of the students who had log files in the system were available. In this data set, the static measure did correlate with the MCAS, with an R-value of 0.71 and the dynamic assistance measure correlates with an R-value of -0.6. Thus, there is some preliminary evidence that the Assistment System may predict student performance on paper-based MCAS items.

It is suspected that a better job of predicting MCAS scores could be done if students could be encouraged to take the system seriously and reduce “gaming behavior”. One way to reduce gaming is to detect it [1] and then to notify the teacher’s reporting session with evidence that the teacher can use to approach the student. It is assumed that teacher intervention will lead to reduced gaming behavior, and thereby more accurate assessment, and higher learning.

The project team has also been exploring metrics that make more specific use of the coding of items and scaffolding questions into knowledge components that indicate the concept or skill needed to perform the item or scaffold correctly. So far, this coding process has been found to be challenging, for instance, one early attempt showed low inter-rater reliability. Better and more efficient ways to use student data to help in the coding process are being sought out. It is believed that as more data is collected on a greater variety of Assistment items, with explicit item difficulty designs embedded, more data-driven coding of Assistments into knowledge components will be possible.

Tracking student learning over time is of interest, and assessment of students using the Assistment system was examined. Given that there were approximately 650 students using the system, with each student coming to the computer lab about 7 times, there was a table with 4550 rows, one row for each student for each day, with an average percent correct which itself is averaged over about 15 MCAS items done on a given day. In Figure 10, average student performance is plotted versus time. The y-axis is the average percent correct on the original item (student performance on the scaffolding questions is ignored in this analysis) in a given class. The x-axis represents time, where data is bunched together into months, so some students who came to the lab twice in a month will have their numbers averaged. The fact that most of the class trajectories are generally rising suggests that most classes are learning between months.

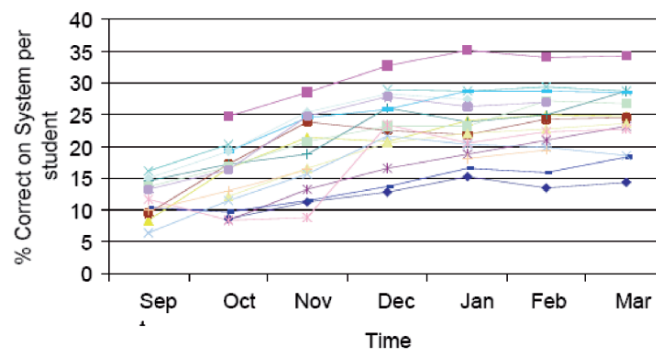


Fig. 2.10. Average student performance is plotted over time.

Given that this is the first year of the Assistment project, new content is created each month, which introduces a potential confounder of item difficulty. It could be that some very hard items were selected to give to students in September, and students are not really learning but are being tested on easier items. In the future, this confound will be eliminated by sampling items

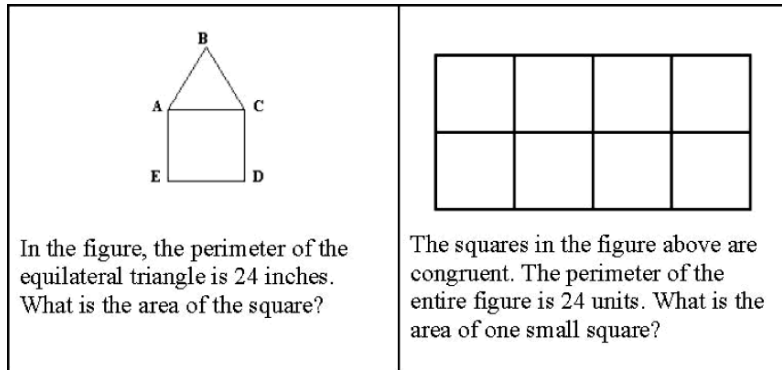
randomly. Adding automated applied longitudinal data analysis [7] is currently being pursued.

#### **2.4.4 Analysis of data to determine whether the system effectively teaches.**

The second form of data comes from within Assistment use. Students potentially saw 33 different problem pairs in random order. Each pair of Assistments included one based on an original MCAS item and a second “morph” intended to have different surface features, like different numbers, and the same deep features or knowledge requirements, like approximating square roots. Learning was assessed by comparing students’ performance the first time they were given one of a pair with their performance when they were given the second of a pair. If students tend to perform better on the second of the pair, it indicates that they may have learned from the instructional assistance provided by the first of the pair.

To see that learning happened and generalized across students and items, both a student level analysis and an item level analysis were done. The hypothesis was that students were learning on pairs or triplets of items that tapped similar skills. The pairs or triplet of items that were chosen had been completed by at least 20 students.

For the student level analysis there were 742 students that fit the criteria to compare how students did on the first opportunity versus the second opportunity on a similar skill. A gain score per item was calculated for each student by subtracting the students’ score (0 if they got the item wrong on their first attempt, and 1 if they got it correct) on their 1st opportunities from their scores on the 2nd opportunities. Then an average gain score for all of the sets of similar skills that they participated in was calculated. A student analysis was done on learning opportunity pairs seen on the same day by a student and the t-test showed statistically significant learning ( $p = 0.0244$ ). It should be noted that there may be a selection effect in this experiment in that better students are more likely to do more problems in a day and therefore more likely to contribute to this analysis. An item analysis was also done. There were 33 different sets of skills that met the criteria for this analysis. The 5 sets of skills that involved the most students were: Approximating Square Roots (6.8% gain), Pythagorean Theorem (3.03% gain), Supplementary Angles and Traversals of Parallel Lines (1.5% gain), Perimeter and Area (Figure 11)(4.3% gain) and Probability (3.5% gain). A t-test was done to see if the average gain scores per item were significantly different than zero, and the result ( $p = 0.3$ ) was not significant. However, it was noticed that there was a large number of negative average gains for items that had fewer students so the average gain scores were weighted by the number of students, and the t-test was redone. A statistically significant result ( $p = 0.04$ ) suggested that learning should generalize across problems. The average gain score over all of the learning opportunity pairs is approximately 2%. These results should



**Fig. 2.11.** A perimeter and area learning opportunity pair.

be interpreted with some caution as some of the learning opportunity pairs included items that had tutoring that may have been less effective. In fact, a few of the pairs had no scaffolding at all but just hints.

### 2.4.5 Experiments

The Assistment System allows randomized controlled experiments to be carried out. At present, there is control for the number of items presented to a student, but soon the system will be able to control for time, as well. Next, two different uses of this ability are described.

#### Do different scaffolding strategies affect learning?

The first experiment was designed as a simple test to compare two different tutoring strategies when dealing with proportional reasoning problems like item 26 from the 2003 MCAS: “The ratio of boys to girls in Meg’s chorus is 3 to 4. If there are 20 girls in her chorus, how many boys are there?” One of the conditions of the experiment involved a student solving two problems like this with scaffolding that first coached them to set up a proportion. The second strategy coached students through the problem but did not use the formal notation of a proportion. The experimental design included two items to test transfer. The two types of analyses the project is interested in fully automating is 1) to run the appropriate ANOVA to see if there is a difference in performance on the transfer items by condition, and 2) to look for learning during the condition, and see if there is a disproportionate amount of learning by condition.

Two types of analyses were done. First, an analysis was done to see if there was learning during the conditions. 1st and 2nd opportunity was treated as a repeated measure and to look for a disproportionate rate of learning due to condition (SetupRatio vs. NoSetup). A main effect of learning between first

and second opportunity ( $p = 0.05$ ) overall was found, but the effect of condition was not statistically significant ( $p = 0.34$ ). This might be due to the fact that the analysis also tries to predict the first opportunity when there is no reason to believe those should differ due to controlling condition assignment. Given that the data seems to suggest that the SetupRatio items showed learning a second analysis was done where a gain score (2nd opportunity minus 1st opportunity) was calculated for each student in the SetupRatio condition, and then a t-test was done to see if the gains were significantly different from zero and they were ( $t = 2.5$ ,  $p = 0.02$ ), but there was no such effect for NoSetup.

The second analysis done was to predict each student's average performance on the two transfer items, but the ANOVA found that even though the SetupRatio students had an average score of 40% vs. 30%, this was not a statistically significant effect.

In conclusion, evidence was found that these two different scaffolding strategies seem to have different rates of learning. However, the fact that setting up a proportion seems better is not the point. The point is that it is a future goal for the Assistment web site to do this sort of analysis automatically for teachers. If teachers think they have a better way to scaffold some content, the web site should send them an email as soon as it is known if their method is better or not. If it is, that method should be adopted as part of a "gold" standard.

### **Are scaffolding questions useful compared to just hints on the original question?**

An experiment was set up where students were given 11 probability items. In the first condition, the computer broke each item down into 2-4 steps (or scaffolds) if a student got the original item wrong. In the other condition, if a student made an error they just got hints upon demand. The number of items was controlled for. When students completed all 11 items, they saw a few items that were morphs to test if they could do "close"-transfer problems.

The results of the statistical analysis were showing a large gain for those students that got the scaffolding questions, but it was discovered that there was a selection-bias. There were about 20% less students in the scaffolding condition that finished the curriculum, and those students that finished were probably the better students, thus making the results suspect. This selection bias was possible due to a peculiarity of the system that presents a list of assignments to students. The students are asked to do the assignments in order, but many students choose not to, thus introducing this bias. This will be easy to correct by forcing students to finish a curriculum once they have started it. For future work, a new experiment to answer this question, as well as several other questions, will be designed and analyzed.

### 2.4.6 Survey of students' attitudes

At the end of the 2004-2005 school year, the students using the Assistment system participated in a survey. 324 students participated in the survey and they were asked to rate their attitudes on statements by choosing Strongly Agree, Agree, Neither Agree nor Disagree, Disagree or Strongly Disagree. The students were presented with statements such as "I tried to get through difficult problems as quickly as possible," and "I found many of the items frustrating because they were too hard." The statements addressed opinions about subjects such as the Assistment system, math, and using the computer.

We wanted to find out what survey questions were correlated with initial percent correct and learning in the Assistment system. The responses to "I tried to get through difficult problems as quickly as possible," were negatively correlated with learning in the Assistment system ( $p = -0.122$ ). The responses to "When I grow up I think I will use math in my job," were positively correlated with learning in the Assistment system ( $p = 0.131$ ). Responses to statements such as "I am good at math," "I work hard at math," and "I like math class," were all positively correlated with students' percent correct in September (at the beginning of Assistment participation).

We believe that the survey results point to the importance of student motivation and attitude in mastering mathematics. For future work, we plan to examine ways to increase student motivation and keep them on task when working on Assistments.

## 2.5 Summary

The Assistment System was launched and presently has 6 middle schools using the system with all of their 8th grade students. Some initial evidence was collected that the online system might do a better job of predicting student knowledge because items can be broken down into finer grained knowledge components. Promising evidence was also found that students were learning during their use of the Assistment System. In the near future, the Assistment project team is planning to release the system statewide in Massachusetts.

## References

1. Anderson, J. R. (1993). *Rules of the mind*. Hillsdale, NJ: Erlbaum.
2. Anderson, J. R., Corbett, A. T., Koedinger, K. R., and Pelletier, R. (1995). Cognitive tutors: Lessons learned. *The Journal of the Learning Sciences*, 4 (2), 167-207.
3. Anderson, J.R., and Pelletier, R. (1991). A development system for model-tracing tutors. In *Proceedings of the International Conference of the Learning Sciences*, 1-8.

4. Baker, R.S., Corbett, A.T., Koedinger, K.R. (2004) Detecting Student Misuse of Intelligent Tutoring Systems. Proceedings of the 7th International Conference on Intelligent Tutoring Systems, 531-540.
5. Campione, J.C., Brown, A.L., and Bryant, N.R. (1985). Individual differences in learning and memory. In R.J. Sternberg (Ed.). Human abilities: An information-processing approach, 103-126. New York: W.H. Freeman.
6. Feng, Mingyu, Heffernan, N.T. (2005). Informing Teachers Live about Student Learning: Reporting in the Assistment System. Submitted to the 12th Annual Conference on Artificial Intelligence in Education 2005, Amsterdam
7. Heffernan, N. T. and Croteau, E. (2004) Web-Based Evaluations Showing Differential Learning for Tutorial Strategies Employed by the Ms. Lindquist Tutor. Proceedings of 7th Annual Intelligent Tutoring Systems Conference, Maceio, Brazil. Pages 491-500.
8. 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.
9. Jarvis, M., Nuzzo-Jones, G. and 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
10. Koedinger, K. R., Aleven, V., Heffernan, T., McLaren, B. and 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. Pages 162-173
11. Koedinger, K. R., Anderson, J. R., Hadley, W. H., and Mark, M. A. (1997). Intelligent tutoring goes to school in the big city. *International Journal of Artificial Intelligence in Education*, 8, 30-43.
12. Livak, T., Heffernan, N. T., Moyer, D. (2004) Using Cognitive Models for Computer Generated Forces and Human Tutoring. 13th Annual Conference on (BRIMS) Behavior Representation in Modeling and Simulation. Simulation Interoperability Standards Organization. Arlington, VA. Summer 2004
13. Mitrovic, A., and Ohlsson, S. (1999) Evaluation of a Constraint-Based Tutor for a Database Language. *Int. J. on Artificial Intelligence in Education* 10 (3-4), pp. 238-256.
14. Mostow, J., Beck, J., Chalasani, R., Cuneo, A., and Jia, P. (2002c, October 14-16). Viewing and Analyzing Multimodal Human-computer Tutorial Dialogue: A Database Approach. Proceedings of the Fourth IEEE International Conference on Multimodal Interfaces (ICMI 2002), Pittsburgh, PA, 129-134.
15. 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.
16. Nuzzo-Jones, G., Walonoski, J.A., Heffernan, N.T., Livak, T. (2005). The eXtensible Tutor Architecture: A New Foundation for ITS. In C.K. Looi, G. McCalla, B. Bredeweg, and J. Breuker (Eds.) Proceedings of the 12th Artificial Intelligence In Education, 902-904. Amsterdam: ISO Press.
17. Razzaq, L., Feng, M., Nuzzo-Jones, G., Heffernan, N.T., Koedinger, K. R., Junker, B., Ritter, S., Knight, A., 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. In C.K. Looi, G. McCalla, B. Bredeweg, and J. Breuker (Eds.) Proceedings of the 12th Artificial Intelligence In Education, 555-562. Amsterdam: ISO Press.
18. Rose, C. P. Gaydos, , A., Hall, B. S., Roque, A., K. VanLehn, (2003), Overcoming the Knowledge Engineering Bottleneck for Understanding Student Language Input , Proceedings of AI in Education.
  19. Singer, J. D. and Willett, J. B. (2003). Applied Longitudinal Data Analysis: Modeling Change and Occurrence. Oxford University Press, New York.
  20. Turner, T.E., Macasek, M.A., Nuzzo-Jones, G., Heffernan, N.T, Koedinger, K. (2005). The Assistment Builder: A Rapid Development Tool for ITS. In C.K. Looi, G. McCalla, B. Bredeweg, and J. Breuker (Eds.) Proceedings of the 12th Artificial Intelligence In Education, 929-931. Amsterdam: ISO Press.