Towards Teachers Quickly Creating Tutoring Systems

by

Michael A. Macasek

A Thesis

Submitted to the Faculty

of the

WORCESTER POLYTECHNIC INSTITUTE

in partial fulfillment of the requirements for the

Degree of Master of Science

in

Computer Science

by

____

Date: December 2005

APPROVED:

____

Professor Neil Heffernan, Thesis Advisor

____

Professor Kathi Fisler, Thesis Reader

____

Professor Michael Gennert, Head of Department

Contents

List of Figures

List of Tables

Abstract

Intelligent Tutoring Systems have historically been shown [7, 8] to be an effective means of educating an audience. While there is great benefit from such systems they are generally very costly to build and maintain. It has been estimated that 200 hours of time is required to produce one hour of Intelligent Tutoring System content [1, 9]. The Office of Navel Research has funding this thesis because they are interested in reducing the cost of construction for Intelligent Tutoring Systems. In order for Intelligent Tutoring Systems to be widely accepted and used in the classroom environment there needs to be a toolset that allows for even the most novice user to maintain and grow the system with minimal cost. The goal of this thesis is to create such a toolset targeted towards the Assistments Project [14]. One of the goals of the Assistments Project is to provide a means for teachers to receive meaningful data from the system that they can take to the classroom environment thus enabling a comprehensive learning solution. The effectiveness of the toolset was measured by its ability to reduce the overall time taken to package and distribute content in an Intelligent Tutoring System by providing the tools and allowing the completion of the tasks to be at a reasonable speed.

1.0 Introduction

Intelligent Tutoring Systems have historically been shown to be an effective means of educating an audience [7, 8]. An Intelligent Tutoring System is an educational tool that presents users with tutors. These tutors provide custom feedback in the form of follow-up questions and/or hints. Intelligent Tutoring Systems are able to track a user's progress and can potentially tailor content towards this known progress. While there is great benefit from such systems they are generally very costly to build and maintain with respect to the tutors and users of the system. Currently it is estimated that at least 200 hours of time is required to produce one hour of Intelligent Tutoring System content [1, 9].

The focus of this thesis work was to create a toolset which enables an Intelligent Tutoring System to be completely managed by teacher users and system administrator users independent of the systems development team. The types of maintenance and management that are of interest relate to the creation/management of tutors and users. It is hoped that these tools will reduce the time spent creating, managing, and maintaining content and users which will allow for a wider adoption of an Intelligent Tutoring System into learning environments. Generally speaking in order for an individual to maintain an Intelligent Tutoring System they would need significant experience in the Computer Science and Cognitive Science fields.

The toolset created under this research has been developed for the Assistment Project [14] where the toolset has been named the Assistment Portal. The Assistment Project is a web based Intelligent Tutoring System currently being developed at Worcester Polytechnic Institute and strives to assess as well as assist the student users of the system. Currently the system is targeted at middle school and high school level mathematics students and used in the Worcester Area Schools in their students preparation for the MCAS [13] exam.

For an Intelligent Tutoring System to be successful the ability of teacher users and system administrator users to self manage the system is a necessity. There is a need for a simple but powerful interface from which teacher users and system administrator users are able to manage all aspects of the system without requiring them to have any Intelligent Tutoring System domain knowledge. Managing the system entails the creation and deployment of content was well as the organization of classes and students in the system.

The Office of Navel Research is funding this work, in part, because they want to reduce the cost of construction for Intelligent Tutoring Systems. Additional funding is being provided by the US Department of Education and the National Science Foundation.

2.0 Background

While this work was primarily oriented towards integration with the Assistments Project many other online tutoring systems exist; many of which provide the similar tutoring functionality as the Assistments Project. The Online Learning Initiative (OLI) [12] from Carnegie Mellon University provides a collection of online tutors directed at many subject areas. While the OLI provides a wide range of online tutors, the tutors lack extensibility to other tutor types and domains, resulting in a high cost for creating content. Cognitive Tutor Authoring Tools

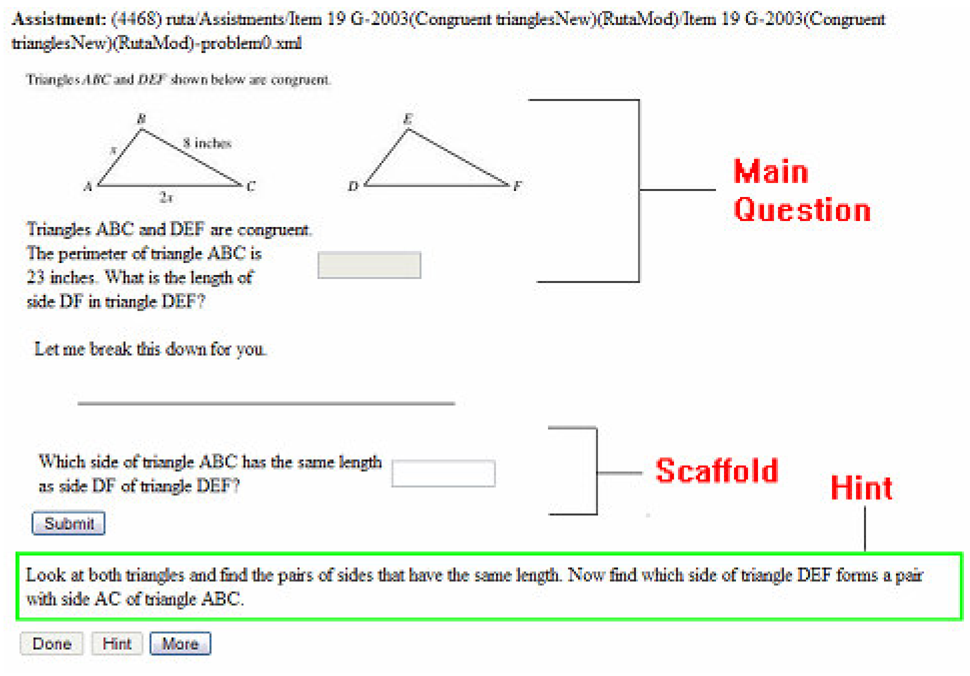
(CTAT) [5], created by LearnLab, also provides online tutoring in addition to being extendable to other domain or content. However, CTAT lacks the administrative tools necessary for non-experts to effectively manage the system. The National Center for Research on Evaluation, Standards and Student Testing (CRESST) [17] provides an online system and has a collection of tools to support the creation and distribution of content. However the CRESST system does not offer tutoring, instead it allows for open ended questions that are then evaluated by a human teacher. SlideTutor [2] is a domain specific web-savvy tutoring system directed at the medical community. This system works well for its domain but was not engineered to be extended to other domains or content.

The Assistment Project has been in existence for about two and a half years and is still under constant development to improve many aspects of the system. Previous to this investigation there were a few tools that allowed the teacher and student users to perform a few basic functions; collectively these tools were colloquially referred to as the Assistment Portal. Initially the functionality was focused on getting the student users in and having them to use the tutoring content. Little attention was paid to developing tools to maintain and grow the system's user base and content.

The original Assistment Portal provided a means to create two types of user accounts: *student* and *teacher* accounts. There did exist a third class of users, *system administrator*s, however there was no way to create a *system administrator* account using the available tools; and once logged into the system, *system administrator*s functioned identically to a teacher user. When a *student* created an account they associated themselves with one teacher's class. Once logged in the student users were only able to see their assignments and begin work on those assignments. An assignment is a collection of tutors that have been made available to student users. When creating a *teacher* user the teacher associated themselves with a school and specified the classes they taught. *Teacher* and *system administrator* users were given access to the Assistment Reports [3], the Assistment Builder [15, 16], and a list of Assistments they created with the ability to preview those Assistments or edit them with the Assistment Builder (Assistment Builder, Assistment Reports, and Assistment will be define in the next paragraphs). Every other maintenance activity was performed by system developers which included, but was not limited to, creating new classes, creating curriculums, assigning/unassigning curriculums, resetting passwords, adding/removing students from a class, looking up usernames, and more. At a later point a limited Curriculum Creator and Curriculum Assigner were also available.

An Assistment (see Figure 1) is one tutor that contains a top-level question which can branch into "scaffolding", hinting strategies, or neither. Hints are simple text only messages provided to guide the user towards the solution. "Scaffolding" questions are follow-up questions that break down the top-level question into its main concepts providing tutoring on the individual concepts. "Scaffolding" questions like the top-level question can contain hints and/or more "scaffolding" questions. Assistments are grouped together into curriculums. There are many types of curriculums supported by the Assistment Project but the most common types are Linear, Random, and Experimental. A Linear Curriculum has a predefined order to the Assistments in the curriculum and presents them to the users in that order where as a Random Curriculum does not have a predefined order instead the Assistments are presented to the user in a random order. The Experimental Curriculum is more advanced and allows for more complex behavior where a

Experimental Curriculum can contain sections that are Linear, Random, or Experimental and only one is selected based on some predefined criteria. Like the Experimental Curriculum both Random and Linear Curriculums can contain other types of sections. Curriculums can then be assigned to a class making them accessible by *student* users.


Assistment: (4468) ruta/Assistments/Item 19 G-2003(Congruent trianglesNew)(RutaMod)/Item 19 G-2003(Congruent trianglesNew)(RutaMod)-problem0.xml

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?

Let me break this down for you.

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

Look at both triangles and find the pairs of sides that have the same length. Now find which side of triangle DEF forms a pair with side AC of triangle ABC.

Figure 1. Assistment with scaffold and hint

The Assistment Builder is a full-featured Assistment creation tool; this tool will be detailed later in the paper. Assistment Reports are also accessible from the Assistment Portal and provide meaningful information to *teacher*s and *system administrator*s related to how well students and classes are performing as well other important information such as problem difficulties. The created curriculums and Assistments are administered to students via the Assistment Runtime [10, 11] which is the underlying framework of the Assistments Project.

In the original Assistment Portal if a *teacher* user wanted to develop and deploy content to their class it would involve a lot of work both on the part of the teacher as well as the Assistment Project development team. The teacher would first begin to develop content with the original Assistment Builder. The original Assistment Builder was constructed as a prototype and proof of concept that was neither complete nor bug free. As a result of these shortcomings users of the original Assistment Builder would often run into problems including the inability to add certain answers or hints to a problem, inability to create scaffolds, and could even have their Assistments deleted or corrupted during editing. It is because of these problems that many times the Assistment Project developers were required to assist the teachers in the creation of their content with most of the cases leading back to having to build the Assistment all over again. Once the teacher and Assistment Project team member had successfully created the Assistments the role of the teacher was complete. From here an Assistment Project developer would have to continue the process. The developer would have to locate the Assistment unique ids in a database. Once the identifiers where found, XML would be manually constructed utilizing the identifiers for the specific type of curriculum to be created. This XML then would be manually inserted into a database and finally the curriculum had to be manually tested to ensure that the XML was well formed. Once the curriculum was created the next task was to manually assign the curriculum to the desired classes. In order for the assignment of the curriculum to be complete the identifiers for the curriculums to be assigned were needed as well as the identifiers for the classes. These identifiers needed to be looked up manually in a database. Once the required identifiers were located the assignment was created by manually inserting the mapping of the curriculum(s) to the desired class(es). Once this mapping was in place the assignment then needed to be manually tested to ensure it was correct. The final step in this process was to then inform the teacher that the assignment was made and ready to be used by students. The important things to notice in the procedure is that at no point was the teacher able to complete a task without the aid of a developer and that most, if not all, of the steps required the developer to manually complete them.

Aside from the creation of content, the management of classes and students was also an important concern of the teachers. During account creation the teacher could specify classes that they taught. This is the only point in which teachers could specify classes and as a result all other user and class management activities had to be completed by a developer. Typically these tasks were verbally communicated to the Assistment Project team. Once the request was received it was placed in a list of management tasks and completed once a developer was available. Tasks of this nature included: creating new classes, resetting passwords, moving students between classes, modifying the name of a class or user, and many more. It is important to notice that there is no way for a teacher to complete any of these tasks as they required a manual change by a developer. Additionally the tasks turn around time was effected by the current work load on developers.

3.0 Methodology

In updating the original tools the entire Assistment Portal was rebuilt from the ground up. Each user type, *student*, *teacher*, and *system administrator*, received a tailored Assistment Portal which contains tools that might be needed for that user type. Recent efforts in the Assistment Project have been made to further reengineer the system into a separate well defined component-based solution to improve its modularity and extensibility [10]. The Assistment Portal was redesigned to echo this new focus and as such each tool was designed independent of the others to allow for the desired modularity. Tools can easily be swapped out of the system with no impact on other tools or overarching functionality of the system. The only drawback in the design of the tools is that they are tied to our system as they utilize our very specific data model. As a result they are not pluggable into any arbitrary system as is; some modification is required to make them fit another systems data model.

This solution was developed with similar technologies to those used in the Assistment Project which include: Apache Tomcat, Apache Ant, SUSE Linux, Oracle, SQL, PL/SQL, Java, JSP, HTML, CSS, JavaScript, DHTML, and AJAX. It was also important to maintain cross web browser compatibility; development sought to maintain compatibility with as many web browsers as possible however the two primary web browsers being supported are: Microsoft Internet Explorer and Mozilla Firefox.

3.1 Information Gathering

During the initial planning of the new Assistment Portal members of the Assistment Project team were consulted in order to decide what functionality was need in the new design. Based on existing teacher input and what members of the team felt were important the following tools were developed: account creation, Assistment Browser, curriculum tools, a modest search engine for searching Assistment Project content, a class/user manager, new login procedure, password management, access to the Assistment Builder, and access to the Assistment Reports. Additionally it was decided to work towards a consistent user interface to help reduce the learning curve for new users.

Once the initial list of features was created it was presented to and discussed with the Assistment Coordinator for Worcester Area Schools to get a sense of how useful and complete the list was. It was suggested that the incorporation of the ability for users to collaborate via the system in a similar fashion that they do in their traditional classroom settings. The collaborations requested included the ability to share resources and results of students using the system. Based on this recommendation it was identified that sharing of Assistments, curriculums, and classes would be desired. It was also noted that some of the tools may be undesired from a teacher viewpoint. However, ultimately, nothing was removed because it was felt that once the functionality was there users could be inclined to use it and find it to be useful.

3.2 Design Goals

The redesigning of the Assistment Portal affected many aspects of the Assistment Project and as a result several design goals were developed to assist in the redesign of the systems core model and ensure that the solution developed was of a certain level of quality. The two main goals of the Assistment Portal were the creation of new tools to replace the manual procedures currently being performed by the Assistment Project developers and to design the tools for novice users so they could complete these tasks unassisted and at a reasonable speed.

The requirements of the Assistment Portal were established for each user type. The *student* user was the most restricted user of the system and had the fewest requirements. *Student* users needed: the ability to create an account, the ability to log into the system, the ability to join classes, the ability to view all classes the student is a member of, the ability to view assignments for a class that the student is a member of, and the ability to start an assignment.

The next level of user, the *teacher* user, allows for significantly greater functionality. *Teacher* user requirements are focused on the development of content and the management of their classes and students. Additionally the there where requirements for: the ability to create an account and the ability to log into the system. The requirements for content creation are broken down into Assistment development and curriculum development. Assistment development requires: the ability to create Assistments, the ability to preview Assistments viewable by that teacher, the ability to share their Assistments, the ability to view certified Assistments, the ability to mark-up Assistments with metadata, and the ability to manage Assistment shares. The requirements for curriculum development were: the ability to create curriculums, the ability to share curriculums, the ability to view curriculums, the ability to mark-up curriculums with metadata, and the ability to assign curriculums to their classes. Management of classes and students for *teacher* users require: the ability to create new classes, the ability to removed old classes, the ability to modify classes (including metadata), the ability to create students in their classes, the ability to remove students from their classes, and the ability to modify a student (including metadata).

The final user type, *system administrator*, shared many of its requirements with the teacher user but included some modifications geared at administration level tasks. The difference in requirements is with respect to user and group management. Since system administrators were in control of every group and user in the system, the restriction to only managing classes was removed as well as the restriction of only creating *student* users.

Tertiary design goals were to ensure system security, extensibility, and modularity. The security part of the design was oriented towards access control. In this web based environment the ability to prevent users from accessing tools they are not allowed to access needed to be in place. Each tool that was developed had its own policy to determine access to an application, based on the type of user attempting to access the tool. Tools are able to identify users as a *student*, *teacher*, or *system administrator* based on their current login. The tools are then responsible for validating if that user is allowed to access it. If the user is not permitted to use that tool they are redirected to that user's main view.

Extensibility and modularity are important for the underlying framework of the Assistment Project because of rapidly changing requirements and revolving project team members. Two main areas of extensibility are in the types of Assistments supported as well as the types of curriculums supported. The type of an Assistment refers to its user interface input type. Currently this includes textbox, checkbox, radio button, and more. The types of curriculums were briefly described above. Moving forward it is easy to see the need for new types of Assistments and curriculums. Even as this paper was being written new types of Assistments were being investigated to provide a better means of tutoring. Modularity is to be supported in that the system contains separated components that can be removed or replaced with no code rewrite on the part of other components, following component-based software engineering practices [4]. Some examples of these components include the Assistment Portal, Assistment Builder, Assistment Runtime, and DataLayer [10]. Although removing some of these components would not make much sense in that it would remove key functionality of the system but replacing them is a foreseeable possibility.

The desire to have certain items in the system shareable, among content creators and teachers, requires the definition of what types of sharing will be allowed. Currently there are two permissions available; *read* and *write*. If a content creator or teacher has a *read* permission for a certain item the user can view the item as well as deploy it to their students (i.e. assigning it to a class). If a user has *write* permissions for a particular item the user then has full access to it. This includes editing the item by adding or removing content from the item. An item is any Assistment, curriculum, class, or student in the system. It is worth exploring other type of permissions in the future including an update type which does not allow for deletion but only corrections and additions to items.

4.0 Implementation

The implementation of the new Assistment Portal required the development/completion of the Database Migration, the Common Tutor Object Platform, and the DataLayer. As such the development of these solutions was incorporated into this thesis work. The creation of the Common Tutor Object Platform and the DataLayer were major tasks unto themselves which required the cooperative effort of three Assistment Project team developers; the author being one of these individuals. Since the Assistment Portal was to rely on these new solutions their implementation has been included to provide a sense of the entire scope of development that was required for the Assistment Portal. The Assistment Portal interacts with the Common Tutor Object Platform to access the Assistment Project content. This content includes Assistments and curriculums.

4.1 Database Migration

Prior to beginning work on these new tools significant changes needed to be put in place. The first of these changes involved to moving of all content into a database. This content included: Assistments, curriculums, and progress files. A progress file indicates the current progress for a given student on a given assignment. The Assistments, curriculums, and progress files were XML files that resided on a file system. The reason for the move was to consolidate all the data associated with the Assistment Project and remove the need to maintain a file system alongside a database.

Initially this move also included the changing of the XML format to a relational format for storing the data. The relational format is better suited for storing the data in a relational database. This involved the process of examining the existing XML and devising a mapping from the XML to a relational table structure. After this phase was completed many weaknesses were discovered mostly involving the choice for the DataLayer (see Section 4.3) and as a result the Assistments, curriculums, and progress files where stored in the database as straight XML strings. The reasons behind this will be discussed in the Discussion section of this paper and serves as an example of the modularity of the new design.

4.2 Common Tutor Object Platform

The Common Tutor Object Platform (CTOP) [10] is the collection of the core objects utilized by the Assistment Project. The CTOP provides a basic component model from which applications and tools can be built. The CTOP is not a full feature component model; as such a replication of existing technology would be redundant and expensive. CTOP does provide some services and features similar to existing component models which allows developers to build their component based Intelligent Tutoring System applications on top of this platform. The core object model consists of a series of components considered to be universally applicable in many different pieces of Intelligent Tutoring System software. These core objects focus on content management and representation, as well as metadata associated with that content. All transaction and write issues are left up to the database's management of such actions.

Content is based in *curriculum* components, which represent a series of *problems*. 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 hierarchy of different types of *sections* and *problems*.

The *section* sub-component is an abstraction for a particular listing of *problems*. This abstraction has been extended to implement the current *section* types, and allows for future expansion of the *curriculum* unit. Currently existing section types include Linear, Random, and Experiment. The *progress* saves a student's state for a given *curriculum* and its *sections*. Also contained within the *progress* is metadata such as total number of *problems* completed and the last updated time.

The *problem* component represents a *problem* to be tutored, including questions and answers required to solve the Assistment. Each of these questions are represented by a *problem* made of two main parts: an *interface* and a *behavior*. The *interface* is interpreted by the Assistment 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* 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 Assistment Runtime composed of “low-level” widgets. “Low-level” widgets are widgets common to many possible platforms of *interface*, and include text labels, text fields, images, radio buttons, etc.

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. Several types of *behaviors* presently exist (state graph tutor, JESS cognitive model), but the interpretation and response to the *behaviors* is up to the application being used. *Behaviors* interact with applications built on the CTOP by producing and using actions. These *actions* are representations of state changes in a specific problem *interface*. The CTOP provides definitions of generic *actions*, as well as actions for each type of *interface* widget. These *actions* provide a messaging layer that allows for communication between components. To allow for scalability and loose coupling of components, these *actions* are XML based and can be passed over a network connection.

Transfer models provide a metadata store of a network of *problems* related to *knowledge components*. This mapping provides a way to track student knowledge over time, as well as a way to organize *problems* in a hierarchal manner with regard to the content of the *problem*. *Transfer models* can be used to provide a model of a student’s knowledge as well as a metric for comparing the value of different *problem* organizational structures.

Finally, there are generic component types, which can be associated with almost every other component in CTOP. These include *properties* and *preferences*, which provide metadata, both time and user specific about specific components or objects.

All of the components described above provide interfaces for their interaction and can thus be easily overridden by a developer. There are also obvious points of coupling where other providers can easily be swapped in and out, such as in the DataLayer, using a variety of methods for persistence. The CTOP provides API's to handle some lifecycle functions, as well as interaction with various components. The DataLayer described above provides an API that provides inflated components of the various types to a consuming application. This API also handles interaction with various component metadata.

An additional API is created by the events generated by problems as actions. The actions are generated by individual interface components and thus are not located in a single entity; however they follow a standard format and can be viewed as an XML service of sorts.

4.3 DataLayer

The DataLayer's function is to separate the Assistment Runtime and Common Tutor Object Platform (CTOP) [10] from storing and retrieving content items. The DataLayer also provides a level of transparency to the CTOP. Components using the CTOP easily access the core objects through the simple DataLayer API, and never worry about storage mechanisms. This allows for different DataLayers that all follow the same API to be easily swapped and CTOP applications can remain unawares. Additionally multiple data sources can be used at the same time, allowing different types of components to be stored in different areas simultaneously. For example, it may be beneficial for some components to be serialized to a relational database, whereas perhaps others would be more effectively stored on a file system.

Each component's interface contains methods that provide access to the object's persistent data. These persistence methods are shared for every instance of that component. For example, every component persists a unique ID, a type, a description, and a link to an interface. The DataLayer uses these methods to create some storable media. The current system creates an XML file that represents the object, and then stores this in a database. It is easily conceivable that this file could also be stored directly onto a file system, or sent across the network to another machine. A previous implementation of the DataLayer used relational persistence to store the object structure a relational database. It did this using the Hibernate tool [5].

4.4 Assistment Portal

With the basic list of desired functionality and underlying architecture in place the next step was to begin grouping the functionality and implement the solutions to meet these goals. The following tools were created: Login, Account Creation, Assistment Browser, Finder, Curriculum Tools, Class Manager, and User Manager. Additionally three separate Assistment Portal views were to be created a *student* view, a *teacher* view, and a *system administrator* view; each tailored to that specific user type's needs. The Assistment Portal is built on top of the CTOP which it uses to access the core objects of the system.

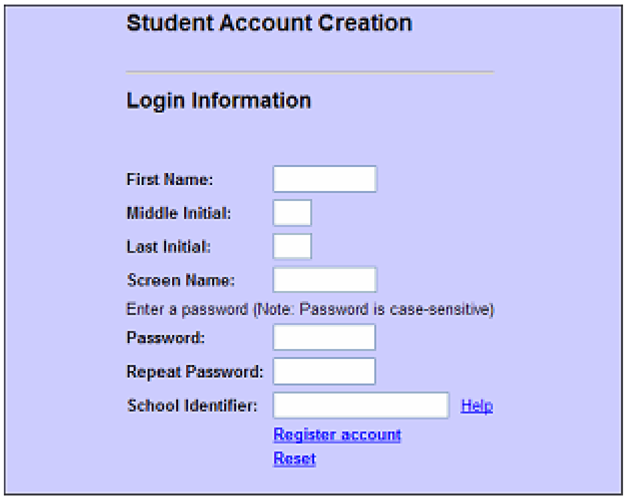
With the tools that have been added there is now significantly more functionality related to the development, deployment, and management of content as well as the added ability for teachers to manage their classes. The new underlying theme found in much of the Assistment

Portal is collaboration. This collaboration is possible between teacher users as well as system administrators throughout the system.

Internal to the Assistment Project, from the software engineering perspective, there was a push to develop pluggable components that operated independently of each other so they could be swapped into and out of the system with ease. Architectural designs of each application were centered on this goal. During the development of these tools several iterations of each tool were produced each of which was reviewed by independent Assistment Project team members and teachers to assess the tools ability to live up to its intended purpose. Through this process each tool was refined to further meet the needs and requirements of the users thus producing a tailored solution.

4.4.1 Account Creation

In order to use the Assistment System an account is required. Users are able to create *student* and *teacher* accounts from the Assistment Project login page. Creating accounts is open to anyone and allowing users to set up an account quickly so they can begin working with the system is a priority.

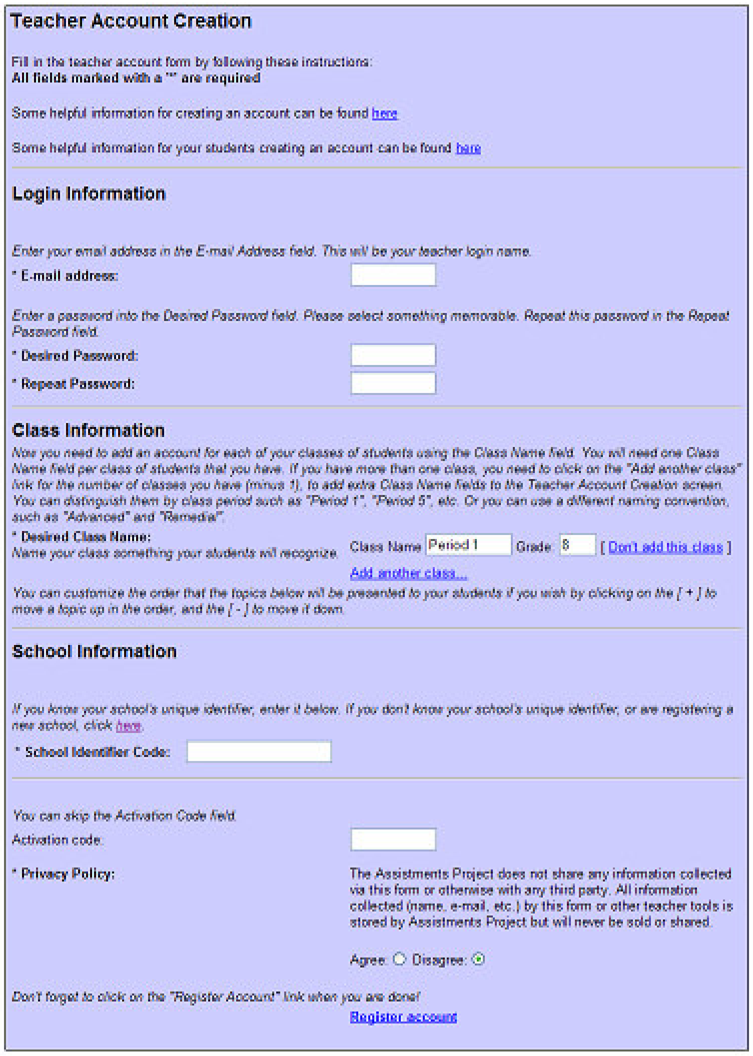

Student Account Creation

Login Information

First Name:
Middle Initial:
Last Initial:
Screen Name:
Enter a password (Note: Password is case-sensitive)
Password:
Repeat Password:
School Identifier:

Figure 2. Student Account Creation

Creating a student account requires very little information, for privacy reasons we do not collect any identifying information about a student (see Figure 2). A new student account requires a first name, middle initial, a last initial, a screen name, an optional password, and the school to which they belong. The screen name is used during login to the system and must be unique over the student's school. For a student to identify a school they must provide the schools unique id which can be typed or looked up via a series of drop down menus. In order for a student to enter a school the school must first exist in the system; generally this occurs during teacher account creation.


Teacher Account Creation

Fill in the teacher account form by following these instructions:
All fields marked with a "*" are required

Some helpful information for creating an account can be found here

Some helpful information for your students creating an account can be found here

Login Information

Enter your email address in the E-mail Address field. This will be your teacher login name.
*** E-mail address:**

Enter a password into the Desired Password field. Please select something memorable. Repeat this password in the Repeat Password field.
*** Desired Password:**
*** Repeat Password:**

Class Information

Now you need to add an account for each of your classes of students using the Class Name field. You will need one Class Name field per class of students that you have. If you have more than one class, you need to click on the "Add another class" link for the number of classes you have (minus 1), to add extra Class Name fields to the Teacher Account Creation screen. You can distinguish them by class period such as "Period 1", "Period 5", etc. Or you can use a different naming convention, such as "Advanced" and "Remedial".
*** Desired Class Name:**
Name your class something your students will recognize. Class Name Period 1 Grade 8 [Don't add this class]
Add another class...
You can customize the order that the topics below will be presented to your students if you wish by clicking on the [+] to move a topic up in the order, and the [-] to move it down.

School Information

If you know your school's unique identifier, enter it below. If you don't know your school's unique identifier, or are registering a new school, click here.
*** School Identifier Code:**

You can skip the Activation Code field.
Activation code:
*** Privacy Policy:** The Assistments Project does not share any information collected via this form or otherwise with any third party. All information collected (name, e-mail, etc.) by this form or other teacher tools is stored by Assistments Project but will never be sold or shared.

Agree: ○ [✓ Disagree: ●]

Don't forget to click on the "Register Account" link when you are done!
Register account

Figure 3. Teacher Account Creation

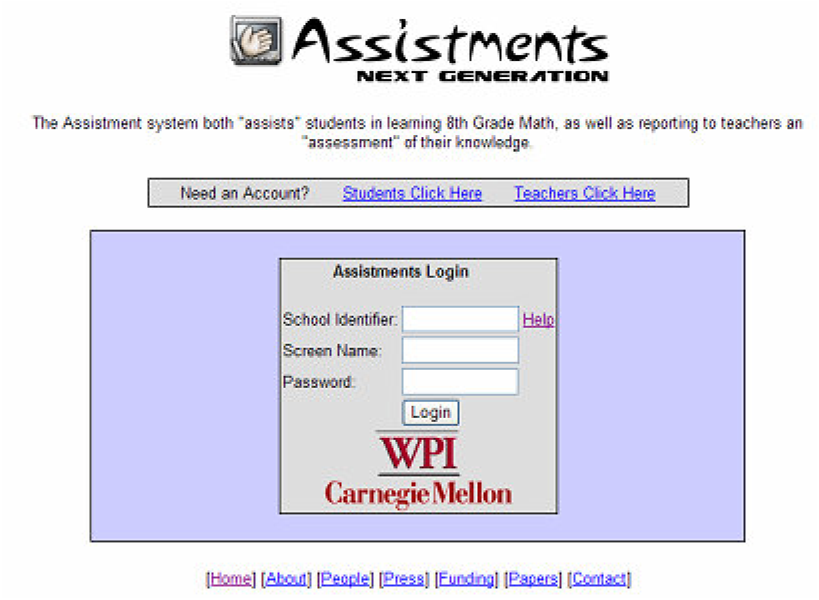
During teacher account creation there is a more involved process in which more information is collected (see Figure 3). A teacher must provide an email address, a password, a viewable teacher name, list of classes and a school association. Teacher accounts are unique over the entire system and their screen names are their email addresses. The viewable teacher name is how the teacher displays their name to look to the students. A teacher has the option of specifying classes during account creation, while this is not the only place classes can be added it is a convenient place to do so. When specifying a class the teacher needs to provide a name for the class as well as grade level for the class. A default sample class is provided for them; it can be changed to better reflect the teachers class names.

A teacher must, in the same way as a student, associate themselves with a school. Unlike students, teachers can create new a new district/town and a new school if the district/town or school they belong to is not in the system. District/Town and classes are part of a hierarchical group structure. At the top of the hierarchy is a state followed by district/town, school, and finally classes. To create a new district/town the teacher first selects a state where the district/town is to be added. In order to specify new district/town the user needs to provide a district/town name that is unique in that state followed by some additional primary contact information. Creating a school is a similar process however the teacher also needs to provide a unique identification string for the school that will be used during login. This identification string must be system unique and is generally a short abbreviation of the school name that is easier to type. Once these new groups are created new students and teachers will be able to join them.

Teacher accounts are not by default activated upon creation. Until a teacher account is activated the account is not usable and students will not be able to join the classes associated with that teacher. There are two ways to activate a teacher's account. First during account creation the user can enter a secret activation code that is not made publicly available. If they do not enter the activation code an email is sent to their entered email address which includes a link to activate their account.

4.4.2 Login

Significant changes have been made with regard to the procedure to log a user into the system. Previously there was a separate login process for student and teacher users. Teachers required a unique user name and password while student users first had to select though a chained pull down system to first locate their state, district, school, and then class. The students needed to select their user name and enter a password. When a student login process began, several posts to the web server were needed to return the information to populate the pull down boxes, look up their unique id, and then finally the login. Since student user names are not unique over the entire system it was necessary to perform a lookup of their globally unique id, based on their school, prior to attempting to authenticate the user. Teacher users did not have this problem as their user names were already unique over the entire system.


Assistments NEXT GENERATION

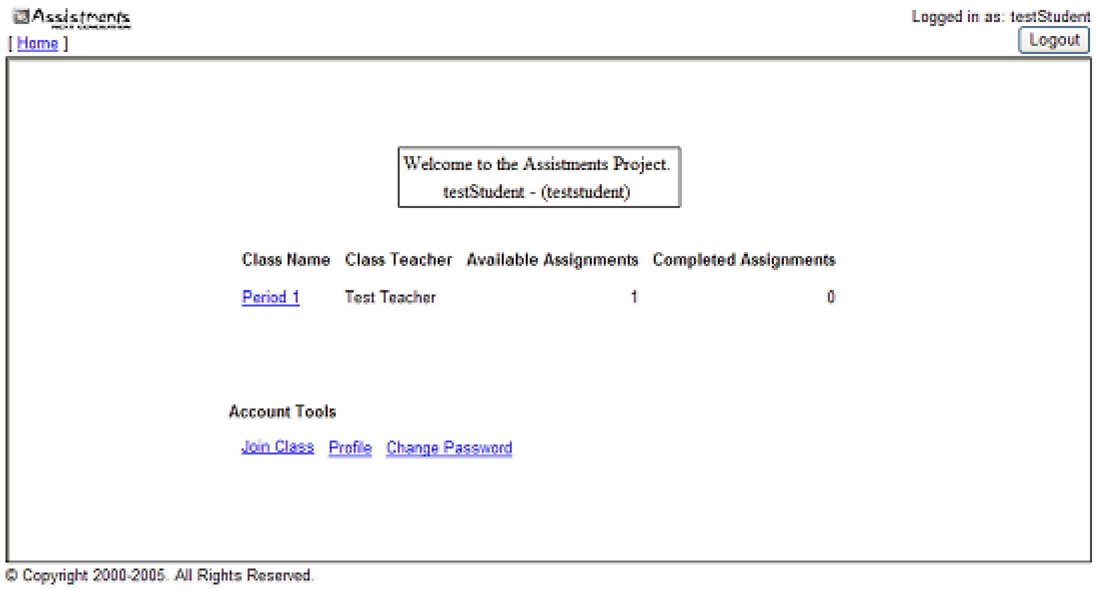
The Assistment system both "assists" students in learning 8th Grade Math, as well as reporting to teachers an "assessment" of their knowledge.

Figure 4. Unified Login System

In an effort to simplify this process the student and teacher login procedure were merged into one method of login (see Figure 4). Each user is required to provide the schools unique id (this can be looked up in a similar fashion to the student account creation), a screen name, and a password. When a user clicks on login there is a lookup of the unique user id based on the screen name and the school entered. While this process is redundant for a teacher user as this id is the same as their screen name, it was decided to be an acceptable cost in order to accomplish the goal of a unified login procedure. Once this lookup is complete the authentication is attempted. When a lookup occurs an asynchronous request is sent to a servlet which performs the lookup and returns the resulting unique identifier; removing the many page requests that were present in the prior implementation. The result, if valid, is passed to the actual login security check. In doing so the number of page posts was reduced to one: the authentication step of login. This lookup is transparent to the user in the new solution and error detection is possible with regard to incorrect usernames and passwords.

4.4.3 Student User

Once a student user has successfully logged into the system they are presented with the student main page (see Figure 5). Previously students only saw a list of assignments. From this student main page a student is able to join a class, update passwords, update their profile, and view a joined class's assignment list. When a new student account is first logged into, a student's first step is to join a class (see Figure 6). This can be done by selecting a class from a teacher/class linked drop down menu. Students are only able to join classes in their school. With this new process students can belong to multiple classes at the same time, where they were previously limited to a single class. Once a student has joined a class they cannot remove themselves from the class, the class teacher must do that. For each class a student belongs to there is a summary of the class assignments, viewable from the student main page, indicating available assignments and completed assignments counts. An assignment is a curriculum that is assigned to a particular class.



Welcome to the Assistments Project.
testStudent - (teststudent)

Class Name	Class Teacher	Available Assignments	Completed Assignments
Period 1	Test Teacher	1	0

Account Tools

Join Class Profile Change Password

© Copyright 2000-2005. All Rights Reserved.

Figure 5. Student User Main View

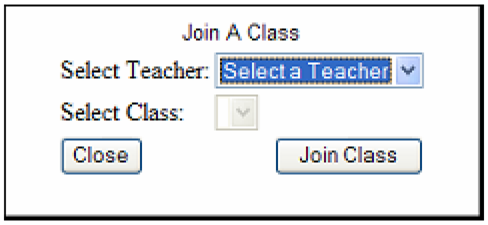
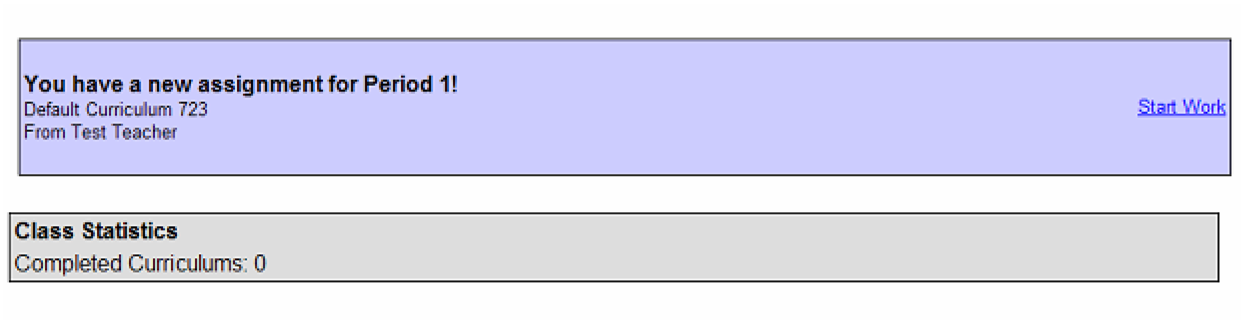


Figure 6. Join A Class

Upon selecting a class from the student's class list the user sees a list of assignments available for this class (see Figure 7). Each assignment indicates whether it has been started "Resume Work" or whether no work was been attempted on this assignment "Start Work". Once an assignment is selected the student enters the Assistment Runtime [11]. Additionally, at the bottom of this page, the user sees a summary indicating how many assignments have been completed for this class.



You have a new assignment for Period 1!
Default Curriculum 723
From Test Teacher

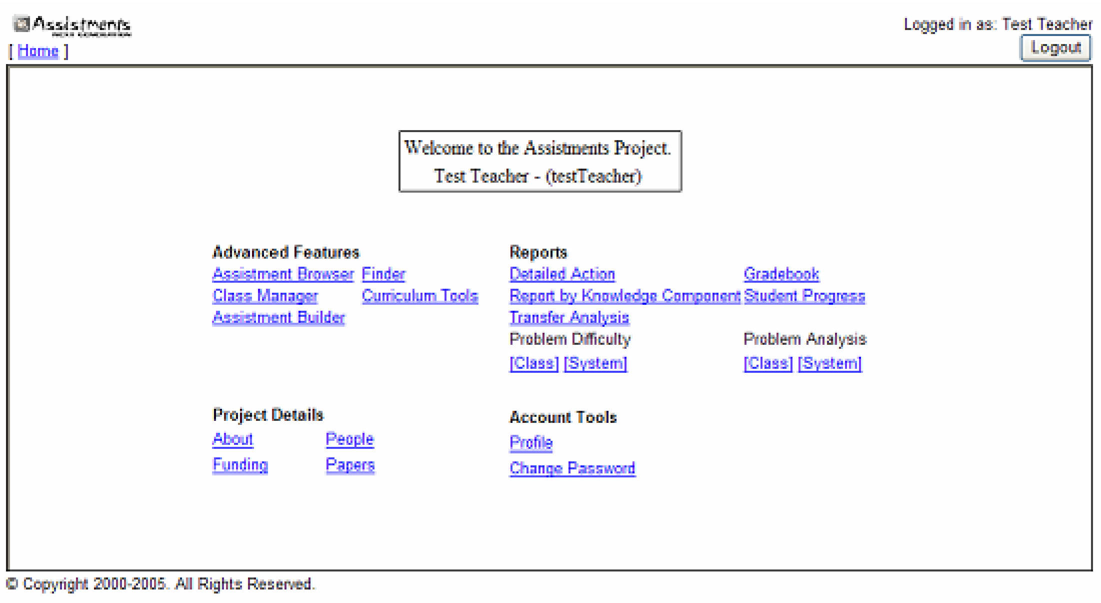
Start Work

Class Statistics
Completed Curriculums: 0

Figure 7. Assignment Listing

4.4.4 Teacher User

Teacher users have a more complicated main page which allows them to access many of the Assistment Portal tools. From the teacher main page, access is provided to the Advanced Features, Reports, Project Details, and Account Tools (see Figure 8). The Advanced Features consist of the Assistment Browser, Finder, Class Manager, Curriculum Tools, and Assistment Builder. The Reports are a separate portion of the Assistment Project, but they are very helpful to teachers, and they have been integrated into this interface for their access. Project Details provide some information related to the Assistment Project and provides some helpful tutorial videos for teachers.


Assistments

[Home]

Logged in as: Test Teacher

Logout

Welcome to the Assistments Project.
Test Teacher - (testTeacher)

Advanced Features
Assistment Browser, Finder, Class Manager, Curriculum Tools, Assistment Builder

Reports
Detailed Action, Gradebook, Report by Knowledge Component, Student Progress, Transfer Analysis
Problem Difficulty [Class] [System]
Problem Analysis [Class] [System]

Project Details
About, People, Funding, Papers

Account Tools
Profile, Change Password

© Copyright 2000-2005. All Rights Reserved.

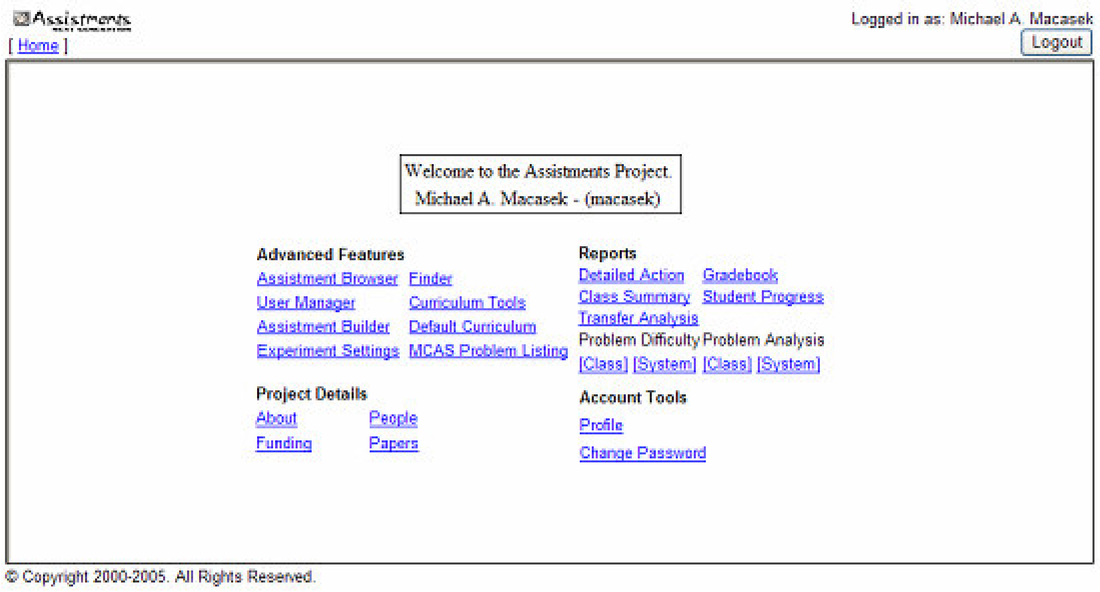
Figure 8. Teacher User Main View

The account tools for a teacher user provide the ability to update a teacher's password as well as manage their profile. Currently the profile consists of some basic information related to the user, such as the teacher's viewable name, first name, middle name, last name, gender as well as some teacher specific information including whether or not the teacher is content area certified and the number of years they have been teaching.

4.4.5 System Administrator User

The system administrator user is currently a catch all group for all Assistment Project team members. Generally speaking there are two types of Assistment Project team users: system developers and content creators. System developers are concerned with system maintenance and stability whereas content creators are concerned with developing Assistments and curriculums. As a result the system administrator main view consists of tools to accomplish both these tasks.

From their main page system administrators have access to Advanced Features, Reports, Project Details, and Account Tools (see Figure 9). Reports, Project Details, and Account Tools are the same tools that are available to teacher users. The Advanced Features contain some of the same tools as the teacher user but also includes several for system administrators. These tools include the Assistment Browser, Finder, User Manager, Curriculum Tools, Assistment Builder, Default Curriculum, Experiment Settings, and MCAS Problem Listing.



Welcome to the Assistments Project.
Michael A. Macasek - (macasek)

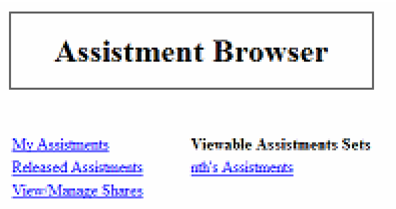
Figure 9. System Administrator User Main View

The Default Curriculum tool allows system administrators to specify what curriculums should be default in every class for a given grade level. When a new class is created the default curriculums are automatically assigned to that class. The Experiment Settings are part of other work currently being investigated under the Assistment Project and this tool allows for the management of controlled experiments [18]. Lastly, the MCAS Problem Listing is a tool that displays a list of all the Massachusetts state certified MCAS problems and the matching Assistments that have been developed to tutor particular problems.

4.4.6 Assistment Browser

The Assistment Browser is a tool used for the manipulation of a user's Assistments (see Figure 10). From the Assistment Browser it is possible to view Assistments, "Released Assistments"; any shared Assistments from other teachers or content creators, and users are able to manage their Assistment Shares. "Released Assistments" are Assistments that have been developed by the Assistment Project team and are viewable by all users in the system. Each of these groups is viewable and depending on the teachers permissions they can perform several actions on the Assistments. Such actions include preview, edit, and marking up of an Assistment with metadata.

Previewing an Assistment allows the viewer to work though the problem continuously verifying its correctness or to discover the content of an Assistment (see Figure 11). The Assistment is viewed by the user in the same way a student user would see the problem while attempting to solve it. If the user's permissions allow for editing, the Assistment can also be loaded into the Assistment Builder and edited.

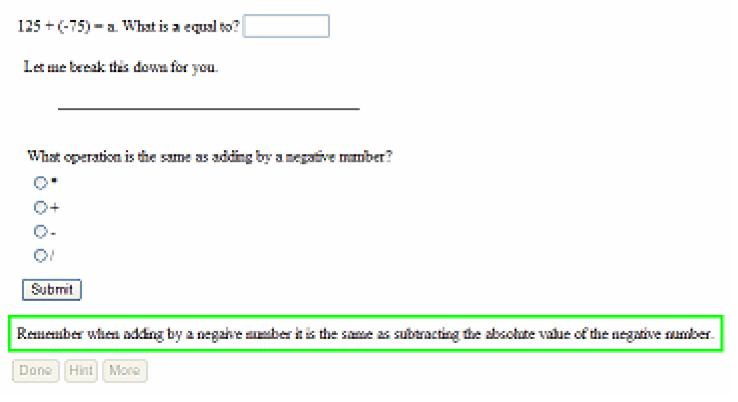


Assistment Browser

My Assistments
Released Assistments
View/Manage Shares

Viewable Assistments Sets
nth's Assistments

Figure 10. Assistment Browser



125 + (-75) = a. What is a equal to?

Let me break this down for you.

What operation is the same as adding by a negative number?

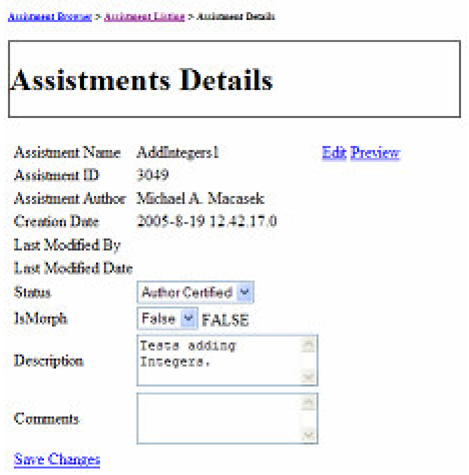
○ *
○ +
○ -
○ /

Submit

Remember when adding by a negative number it is the same as subtracting the absolute value of the negative number.

Figure 11. Assistment Preview

The Assistment Browser provides an interface to add metadata to an Assistment (see Figure 12). Some metadata is automatically associated with an Assistment such as author, date/time created, date/time modified, and the user that performed the modification. In addition to this automatically tracked data it is also possible to specify if this Assistment is a “morph”, the status of the Assistment, a brief description, and brief comments about the Assistment. A “morph” is a term used to describe an Assistment that is similar in setup or content to another Assistment but with different data and answers. A “morph” is generally created by loading an existing Assistment into the Assistment Builder, altering the content, and then saving the Assistment under a new name.



Assistment Browser > Assistment Listing > Assistment Details

Assistments Details

Assistment Name	AddIntegers1	Edit Preview
Assistment ID	3049	
Assistment Author	Michael A. Macasek	
Creation Date	2005-8-19 12:42:17.0	
Last Modified By		
Last Modified Date		
Status	Author Certified	
IsMorph	False FALSE	
Description	Tests adding Integers.	
Comments		

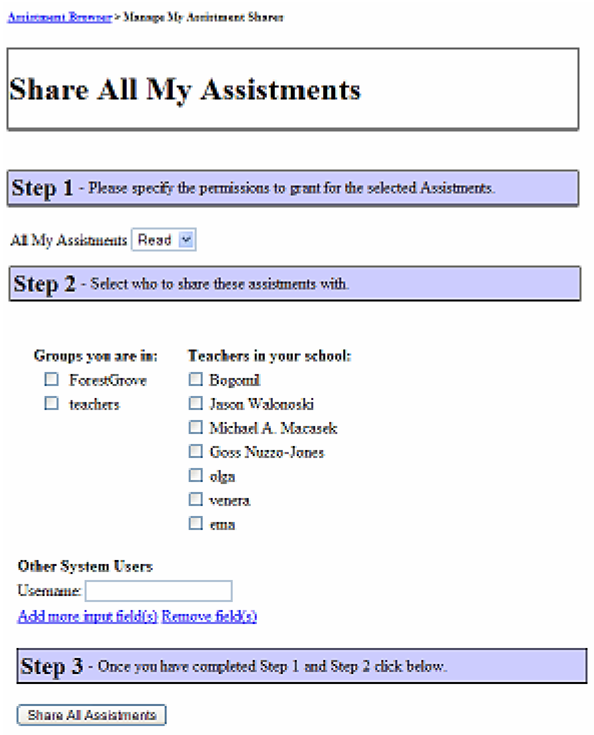
Save Changes

Figure 12. Assistment Details

Assistments can have varying levels of status which indicate the current state of development and quality of the items. Currently there are five status types; Draft, Ready to Test, Author Certified, WPI Certified, and Corrupt. When an Assistment is under development its status is set to Draft, indicating that it is not yet complete. Once an Assistment is believed to be complete, the content creator can change the status to Ready to Test. Ready to Test indicates that the majority of the development is complete for this Assistment and it should be tested to verify its completeness. Once everything has been reviewed the Assistment author can set the status to Author Certified and the Assistment is ready to be used. The WPI Certified status denotes that an item is perceived to be of high quality and that it should be included in the "Release Assistments" list which is viewable by every teacher and content creator. If an Assistment is marked Corrupt then there is a problem with the data related to this Assistment in our database; typically this is an indication of malformed XML. A tool to repair corrupt Assistments was created and is only accessible by system administrators as it allows for the direct access of the Assistments XML; this tool will be briefly discussed later in this document.

Assistments can be shared via the Assistment Browser with any teacher or content creator in the system. Sharing allows collaboration between users in the development and deployment of content. When creating an Assistment Share the user can choose from two levels of permissions: *read* and *write*. A user has inherent *write* privileges to their own Assistments, "Release Assistments" are available as *read* only. When establishing a share the owner can specify the privilege to grant to a user or group of users.

Assistment Shares are created in two ways. An Assistment author can share all their Assistments (see Figure 13) or specific Assistments can be selected to share. It is important to note that if an author elects to share all their Assistments all newly created items will belong to this share as well. The advantage here is that the author does not have to specify a new share after they have created a new item. When creating the share the author will have the option to set the permission level for the Assistments in the share.


Assistment Browser > Manage My Assistment Shares

Share All My Assistments

Step 1 - Please specify the permissions to grant for the selected Assistments.

All My Assistments: Read

Step 2 - Select who to share these assistments with.

Groups you are in:
- ☐ ForestGrove
- ☐ teachers

Teachers in your school:
- ☐ Bogomil
- ☐ Jason Walonoski
- ☐ Michael A. Macasek
- ☐ Goss Nuzzo-Jones
- ☐ olga
- ☐ venera
- ☐ ema

Other System Users
Username:
Add more input field(s) Remove field(s)

Step 3 - Once you have completed Step 1 and Step 2 click below.

Figure 13. Creating an Assistment Share

When sharing all Assistments the user must select one permission level for all their Assistments. If a user is creating a share for selected Assistments, permission levels are set for each Assistment in the share. To add users to this new share, the creator of the share has the option of allowing access to two preexisting groups: all teachers and the author's school. All teachers will allow every teacher in the system to view this share whereas the author's school will allow only teacher in the author's school to view the share. In addition to these two predefined groups, a share can be provided to any group of users that the author would like to share with. When creating this new share group the author has the opportunity to share with teachers in the same school by checking off their name or by entering a username into a text field. By allowing a user to enter usernames into a text field it is possible for a share to be constructed with any users in the system; the only restriction is that the user must know the username of the individual.

When the share is created a new group is automatically created which includes all the users that were specified for the share. These groups are not directly accessible by system users but serves as a means to identify individuals in shares. A user can manage their Assistment Shares from the Assistment Browser. Users can view who is in a share as well as what Assistment this share allows them to see, along with the granted permission levels. At any time the share creator can remove a user from the share preventing further access to those Assistments. If a user is in multiple shares which all include the same Assistment the permission granted to this user is the highest level granted, thus if a user has *read* permission in one share and *write* permission in another share for the same Assistment the user is granted *write* permission for the Assistment.

4.4.7 <u>Finder</u>

The Finder is a modest search engine for the Assistment Project which allows users of the system to locate many types of data for which they have access privileges (see Figure 14). This includes classes, Assistments, curriculums, students, teachers, and more. Users can search over the entire system or restrict their search to particular types of data.

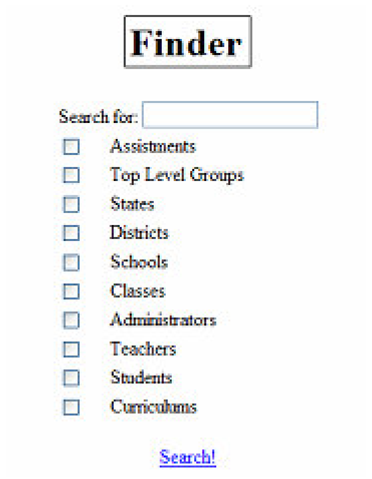
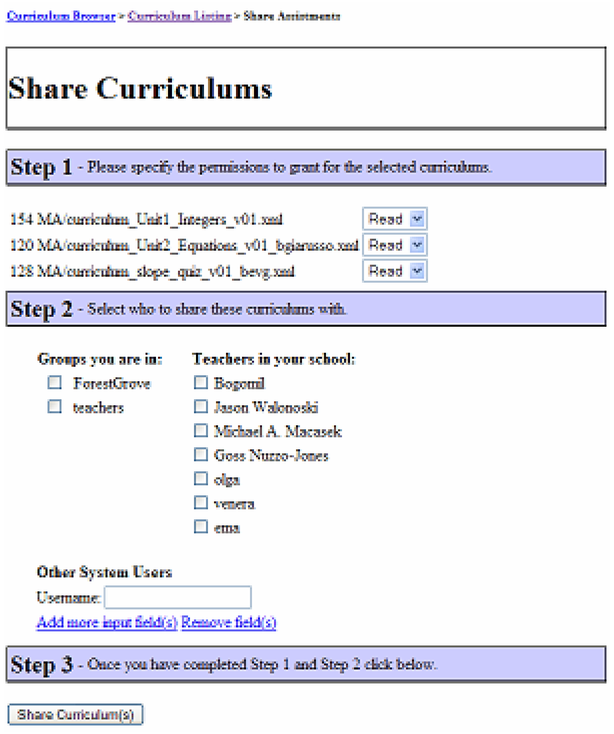


Figure 14. Finder

Special attention is paid to ensure that when searching, only data that the user has been granted access to is searched and returned. When a search is performed the system will examine mostly the metadata in order to find a match for a query. When results are returned to the user they are presented in categories depending on the type of data being search. This is to say that if the user searched over Assistments and curriculums the results will be broken down into two groups: results from Assistments and results from curriculums. The result list offers the user the ability to open an item via its appropriate tool for viewing and manipulation. If the result is an Assistment it can be viewed via the Assistment Browser, if the result is a curriculum is can be viewed via the Curriculum Tools, and so on. This search tool is useful when a teacher who is part of many Assistment Shares is interested in viewing all the questions they are able to view related to a certain concept, that they can easily use the Finder to locate relevant items.

4.4.8 Curriculum Tools

The Curriculum Tools are designed to deploy content to students. With this tool a teacher or content creator can view their curriculums, view "Released Curriculums", view shared curriculums, create new curriculums, edit existing curriculums, and assign curriculums to their classes. Curriculums can also be shared in much the same way as Assistments are shared. The Curriculum Shares have two permission levels; *read* and *write*, they can also be shared in whole or in part just like Assistments Shares. Management of Curriculums Shares is the same as well (see Figure 15).


Curriculum Browser > Curriculum Listing > Share Assistments

Share Curriculums

Step 1 - Please specify the permissions to grant for the selected curriculums.

154 MA/curriculum_Unit1_Integers_v01.xml	Read
120 MA/curriculum_Unit2_Equations_v01_bgiarusso.xml	Read
128 MA/curriculum_slope_quiz_v01_bevg.xml	Read

Step 2 - Select who to share these curriculums with.

Groups you are in:
- ☐ ForestGrove
- ☐ teachers

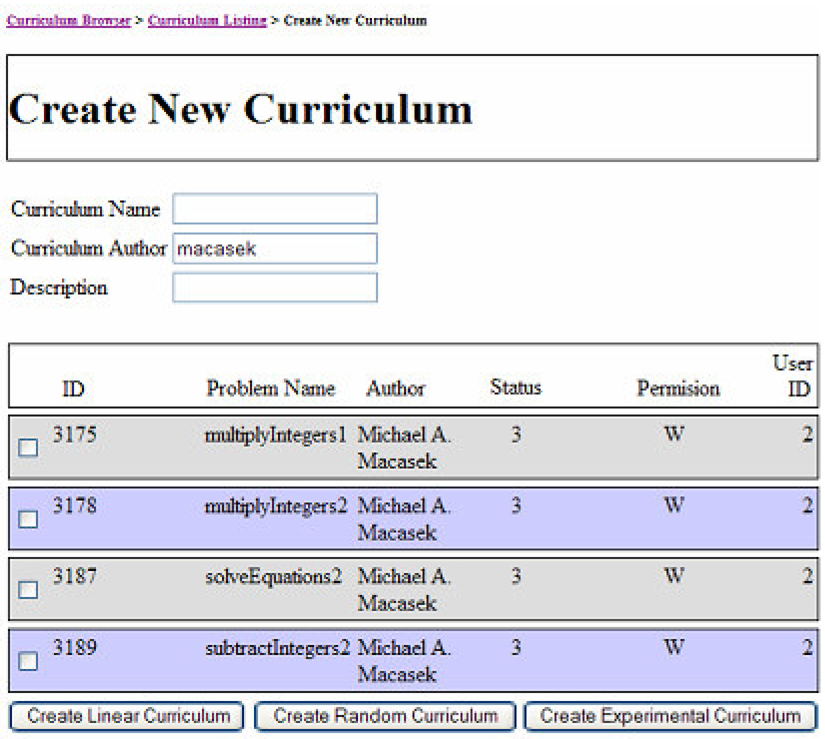
Teachers in your school:
- ☐ Bogomil
- ☐ Jason Walonoski
- ☐ Michael A. Macasek
- ☐ Goss Nuzzo-Jones
- ☐ olga
- ☐ venera
- ☐ ema

Other System Users
Username:
Add more input field(s) Remove field(s)

Step 3 - Once you have completed Step 1 and Step 2 click below.

Figure 15. Creating a Curriculum Share

There are several types of curriculums that the Assistment Project supports. The Curriculum Tools allows for the creation of the most common types: Linear, Random, and Experimental. Curriculums can contain Assistments from the users created Assistments, "Released Assistments", and shared Assistments. When creating a curriculum (see Figure 16) the user selects the Assistments to be included in the curriculum and specifies the curriculum type to create. A name and brief description must also be provided.


Curriculum Browser > Curriculum Listing > Create New Curriculum

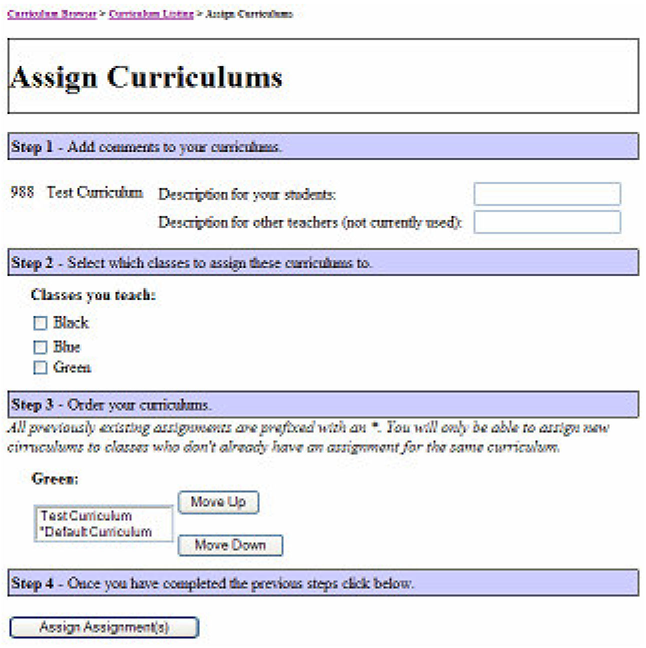
Create New Curriculum

Curriculum Name	
Curriculum Author	macasek
Description	

ID	Problem Name	Author	Status	Permision	User ID
☐ 3175	multiplyIntegers1	Michael A. Macasek	3	W	2
☐ 3178	multiplyIntegers2	Michael A. Macasek	3	W	2
☐ 3187	solveEquations2	Michael A. Macasek	3	W	2
☐ 3189	subtractIntegers2	Michael A. Macasek	3	W	2

Figure 16. Curriculum Creation

A teacher can assign curriculums (see Figure 17) to their classes by selecting the curriculum(s) to assign and specifying the class(es) to for which the curriculum(s) will be assigned. The user also provides a description for the assignment that the students will be able to see; the description should describe the content of the curriculum. The final step in assigning a curriculum is to specify the order of the curriculum with respect to any other assignments the class may have. This order will reflect in the order the student will see the curriculums listed on the assignment page. Once the curriculum is assigned it is then available to the class's students, and it can then be worked on. Curriculums created by that teacher, from "Release Curriculums", and from shared curriculums can be assigned to the user's classes.



Curriculum Browser > Curriculum Listing > Assign Curriculums

Assign Curriculums

Step 1 - Add comments to your curriculums.

988	Test Curriculum	Description for your students:	
		Description for other teachers (not currently used):	

Step 2 - Select which classes to assign these curriculums to.

Classes you teach:

- ☐ Black
- ☐ Blue
- ☐ Green

Step 3 - Order your curriculums.

*All previously existing assignments are prefixed with an *. You will only be able to assign new cirruculums to classes who don't already have an assignment for the same curriculum.*

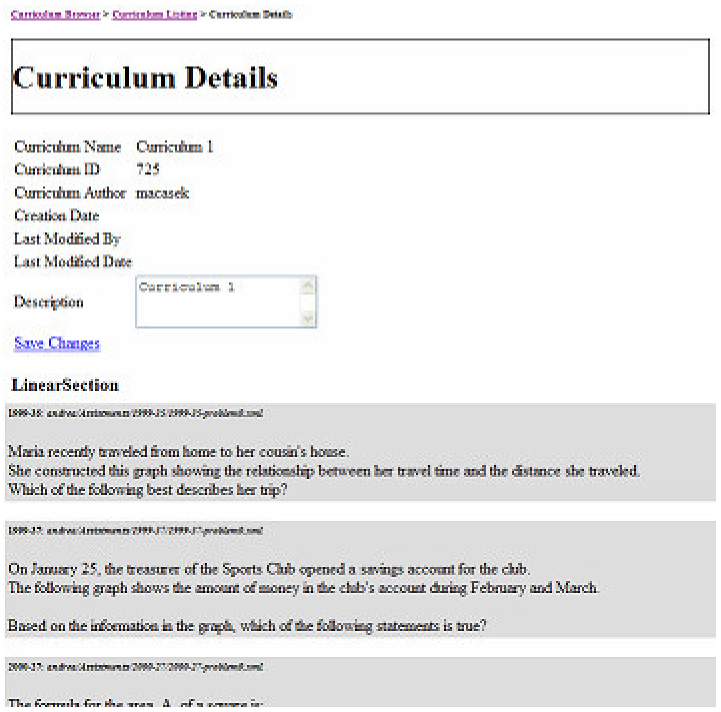
Green:

Test Curriculum
*Default Curriculum

Step 4 - Once you have completed the previous steps click below.

Figure 17. Curriculum Assignment

Curriculums can also be previewed and with their associated metadata. When a curriculum is previewed the viewers are able to see the Assistments in the curriculum; the viewers are not able to progress though the curriculum as a student would (see Figure 18). Most of the metadata associated with a curriculum is automatically generated and includes author, date/time created, date/time modified, and modified by user. Additionally a user can specify a description for the curriculum to better describe its contents.



Curriculum Browser > Curriculum Listing > Curriculum Details

Curriculum Details

Curriculum Name	Curriculum 1
Curriculum ID	725
Curriculum Author	macasek
Creation Date	
Last Modified By	
Last Modified Date	
Description	Curriculum 1

LinearSection

1999-18: andrea/Assistments/1999-15/1999-15-problem8.xml

Maria recently traveled from home to her cousin's house.
She constructed this graph showing the relationship between her travel time and the distance she traveled.
Which of the following best describes her trip?

1999-27: andrea/Assistments/1999-27/1999-27-problem8.xml

On January 25, the treasurer of the Sports Club opened a savings account for the club.
The following graph shows the amount of money in the club's account during February and March.

Based on the information in the graph, which of the following statements is true?

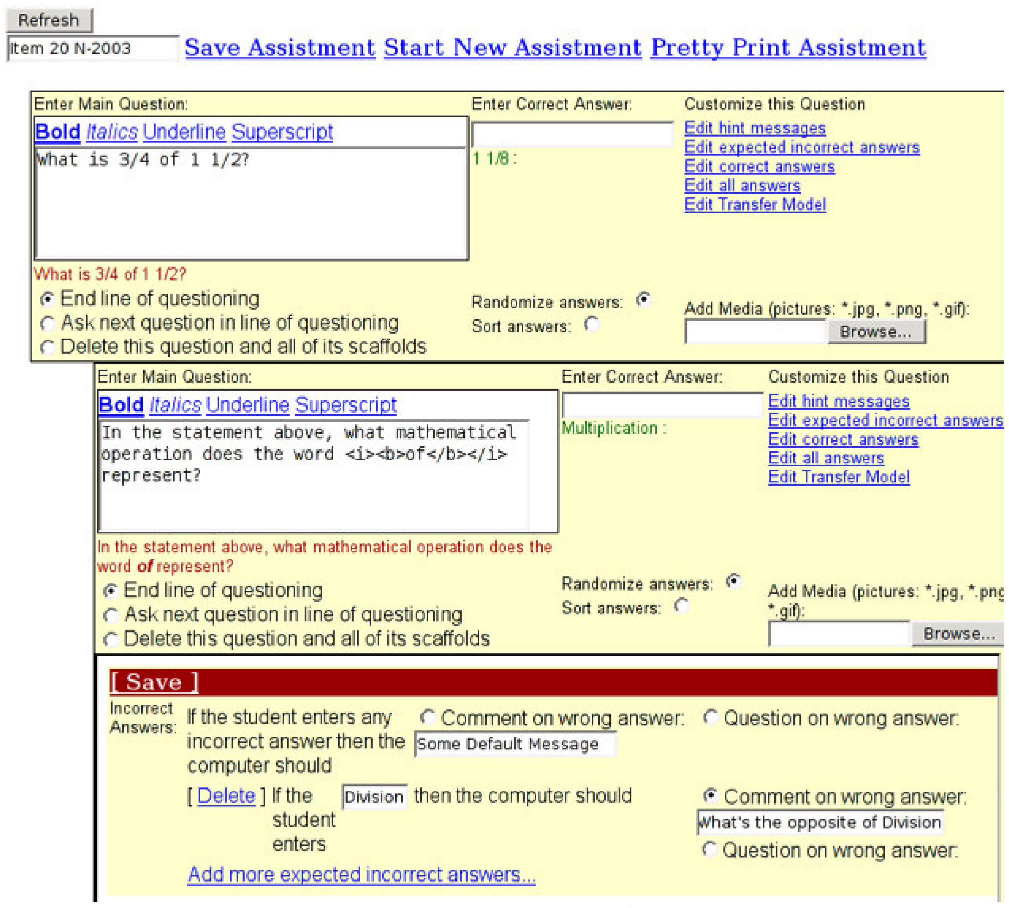
2000-27: andrea/Assistments/2000-27/2000-27-problem8.xml

The formula for the area, A, of a square is:

Figure 18. Previewing a Curriculum

4.4.9 Assistment Builder

The Assistment Builder [15, 16] is a web application designed for accessibility and ease of use, so that a teacher or content creator can create, test, and deploy an Assistment without installing any additional software. If further changes or editing are needed, the Assistment can be loaded into the Assistment Builder, modified, and then redeployed across all the curriculums that make use of the tutor. By making the Assistment Builder available over the web, if a new feature is added the users do not need to update any software. They reap the benefits of any changes made to the system as soon as they log on.


Refresh
Item 20 N-2003
Save Assistment Start New Assistment Pretty Print Assistment

Enter Main Question:
Bold Italics Underline Superscript
What is 3/4 of 1 1/2?
What is 3/4 of 1 1/2?
[✓ End line of questioning]
Ask next question in line of questioning
Delete this question and all of its scaffolds

Enter Correct Answer:
1 1/8 :
Randomize answers: [✓]
Sort answers:

Customize this Question
Edit hint messages
Edit expected incorrect answers
Edit correct answers
Edit all answers
Edit Transfer Model
Add Media (pictures: *.jpg, *.png, *.gif):
Browse...

Enter Main Question:
Bold Italics Underline Superscript
In the statement above, what mathematical operation does the word <i><b>of</b></i> represent?
In the statement above, what mathematical operation does the word ***of*** represent?
[✓ End line of questioning]
Ask next question in line of questioning
Delete this question and all of its scaffolds

Enter Correct Answer:
Multiplication :
Randomize answers: [✓]
Sort answers:

Customize this Question
Edit hint messages
Edit expected incorrect answers
Edit correct answers
Edit all answers
Edit Transfer Model
Add Media (pictures: *.jpg, *.png *.gif):
Browse...

[Save]
Incorrect Answers: If the student enters any incorrect answer then the computer should
Comment on wrong answer: Question on wrong answer:
Some Default Message
[Delete] If the student enters Division then the computer should
[✓ Comment on wrong answer:]
What's the opposite of Division
Question on wrong answer:
Add more expected incorrect answers...

Figure 19. The Assistment Builder - Initial Question, One Scaffold, and Incorrect Answer View

When a user first begins to use the Assistment Builder they are greeted by the standard blank skeleton question. The initial blank skeleton question is used to create the root question. The user enters 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.

Initially the user is presented with the *Main View* (see Figure 19). In this view question text, correct answers, and images can be added to the question. Additionally the user can add new scaffolding off 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*. This can be done by selecting "Edit all answers" from the list of links below Customize this Question in the *Main View*. In the *All Answers View* the user has the option to add additional correct answers as well as expected incorrect answers. The expected incorrect answers allow a teacher to specify the answers students will most likely expect as the correct answer and provide feedback in the form of a message or scaffolding.

Once the user has finished with this view either the *Correct Answer View* or the *Incorrect Answer View* (see Figure 19) is the next step. In the *Correct Answer View* the user can now provide a specific message to be displayed on a correct answer as well as add additional correct answers. From the *Incorrect Answer View* further information can be provided as to the action that will be taken on a specific incorrect answer; new incorrect answers can also be added here. The user now has the 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.

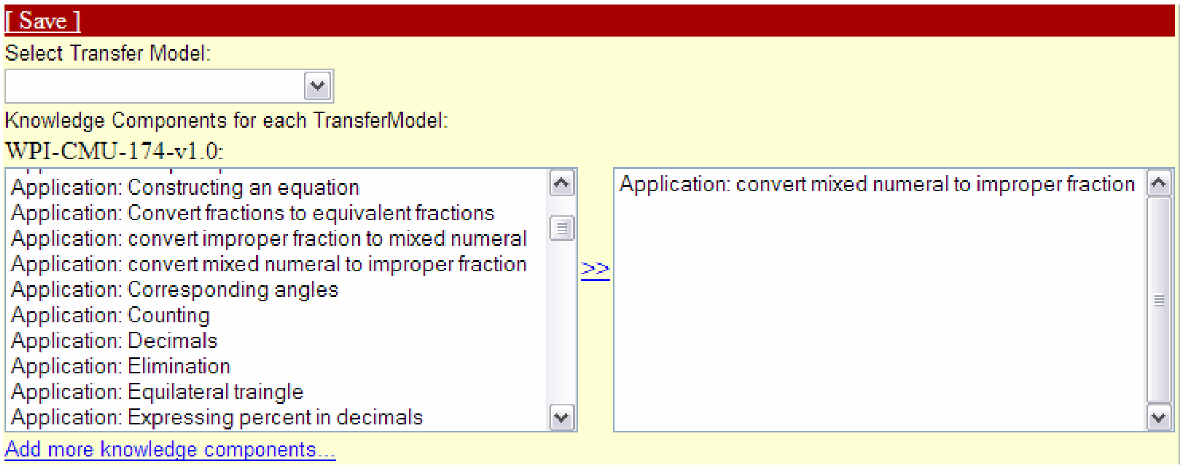

[Save]
Select Transfer Model:
Knowledge Components for each TransferModel:
WPI-CMU-174-v1.0:
Application: Constructing an equation
Application: Convert fractions to equivalent fractions
Application: convert improper fraction to mixed numeral
Application: convert mixed numeral to improper fraction
Application: Corresponding angles
Application: Counting
Application: Decimals
Application: Elimination
Application: Equilateral traingle
Application: Expressing percent in decimals
>>
Application: convert mixed numeral to improper fraction
Add more knowledge components...

Figure 20. Transfer Model View

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 20). In this view the user has the option of specifying one or more skills that are associated with this particular question.

It is important to note that each view, except the *Main View*, requires the user to save updates before leaving the view. This ensures that all entered data is stored. At any point the user can preview the problem by clicking on the preview link to view the current state of the question.

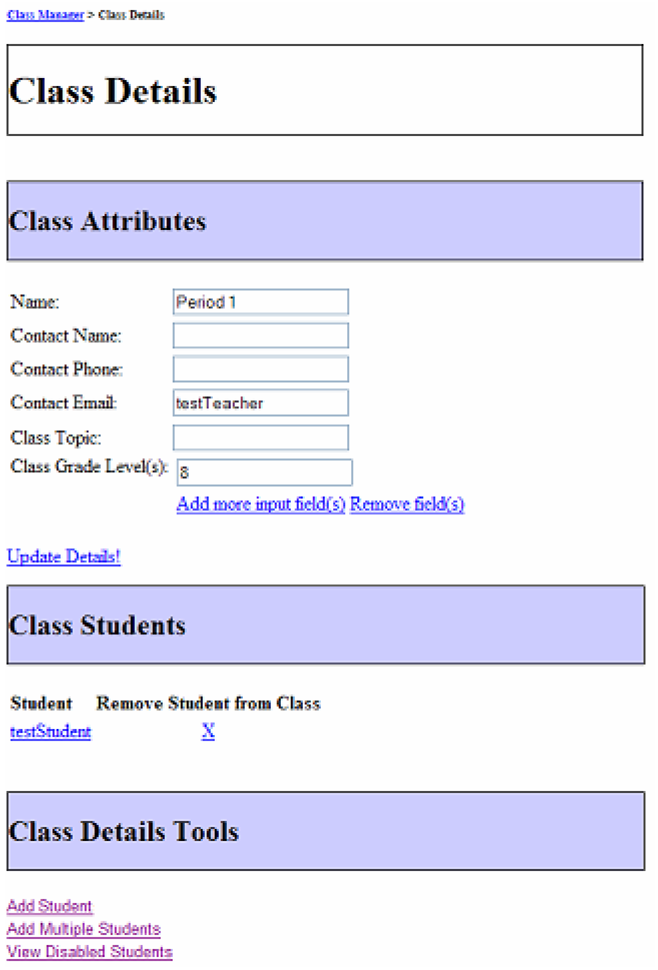
As mentioned above there are two methods of providing scaffolding questions; either by selecting Ask Next Line of Questioning from the *Main View* or specifying scaffolding on a specific incorrect answer. In utilizing either of these methods a new skeleton question will be inserted into the correct position below the current question. Creating a scaffolding question is done in the exact manner as the root question. Once a user is satisfied with the root question and its scaffolding they provide a name for the question and save the *Assistment*. After saving the *Assistment* the tutor is ready to be used assuming that it is complete. An *Assistment* can be modified at any time by loading it into the Assistment Builder and changing its properties as desired.

It is important to note that if content is modified to a great extent it is possible that this will affect the Assistment Reporting on these Assistments. This requires significant attention, moving forward, in order to assess the level to which editing an Assistment can affect its reporting. Along with this, allowing for "versioning" of Assistments and other Assistment Project related content could assist in accurately reporting on Assistments.

The Assistment Builder is currently being redesigned to better serve the users and utilize the Assistment Project core object architecture [10]. The preceding description is based on the initial Assistment Builder; designed in part by this author.

4.4.10 Class/User Manager

The Class Manager tool allows a teacher to manage their classes and students, and control who else can view the class information. Adding, modifying, and removing classes are useful operations for year-to-year changes in schedules. Student management allows a teacher to add a new student into the system and to a particular class. Additionally, if a student has accidentally signed up for a class or has transferred out of a class the teacher can remove them from the class. Students and classes can have additional metadata associated with them to further describe the nature of a class or the abilities of a student. For a class, teachers can specify the class topic and the grade level(s) of the class (see Figure 21). For students, teachers can specify: birth date, student identification number, gender, if the student receives free lunch, is low income, is special education, is English proficient, and the ethnicity of the student (see Figure 22); none of this information has been made searchable at this point. Teachers also have the ability to reset a student's password in the event that a student forgets their password.



Class Manager > Class Details

Class Details

Class Attributes

Name:	Period 1
Contact Name:	
Contact Phone:	
Contact Email:	testTeacher
Class Topic:	
Class Grade Level(s):	8
	Add more input field(s) Remove field(s)

Update Details!

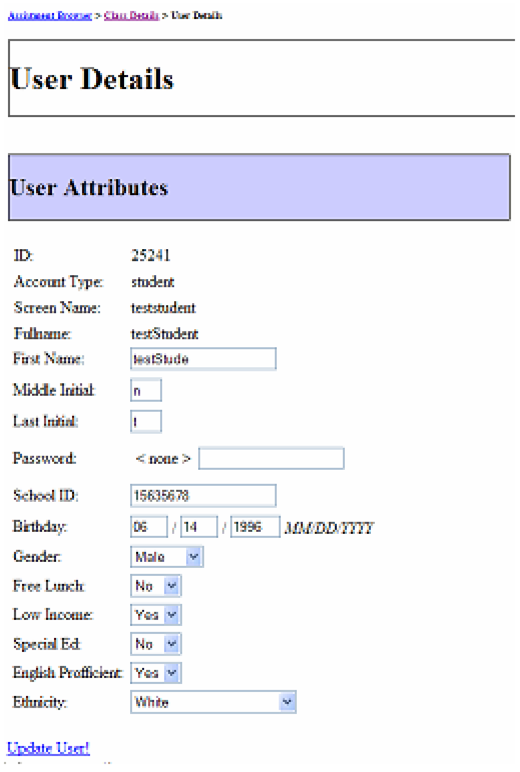
Class Students

Student	Remove Student from Class
testStudent	X

Class Details Tools

Add Student
Add Multiple Students
View Disabled Students

Figure 21. Class Details



Assistment Browser > Class Details > User Details

User Details

User Attributes

ID:	25241
Account Type:	student
Screen Name:	teststudent
Fullname:	testStudent
First Name:	testStude
Middle Initial:	n
Last Initial:	t
Password:	< none >
School ID:	15635678
Birthday:	06 / 14 / 1996 MM/DD/YYYY
Gender:	Male
Free Lunch:	No
Low Income:	Yes
Special Ed:	No
English Profficient:	Yes
Ethnicity:	White

Update User!

Figure 22. User Details

Classes, much like Assistments and curriculums, are sharable to other teachers in the system. This feature will allow teachers to share classes with teacher aids and/or supervisors. With class sharing, a school departmental structure could be mimicked to allow appropriate supervisors access to classes, making them able to assist in the management of the class and reviewing of the student's progress via Assistment Reports. Class Shares are granted with only one permission level; *write*. Establishing shares is done in a similar manner as Assistments and curriculums.

The User Manager is a tool only for system administrator which allows browsing of all system groups and users. This tool is designed to help administrators manage users without the need to access the database directly. It allows for creating of new groups and users of any level as well as modifying existing groups and users. The User Manager does not have a sharing feature as anyone with access to the User Manager can already see everything.

4.4.11 PortalLogger

Since the Assistment Portal has now grown into a mature application the need to log user actions is desired to assist in the evaluation of the tools as well as access each tools usage.

Logging user’s actions can provide information regarding system usage, application evaluation, and possibly reveal any shortcoming with the tools. The PortalLogger records all actions taken by an Assistment Portal user and records them in a database.

The PortalLogger logs the action type, user id, date/time of the action, web browser session id, IP address, and the action details. The action type is a standard type defined for each action in the Assistment Portal and provides a general sense of what action was performed. Action details are a nonstandard string that represents what action just took place. For each action type there is a defined expected action detail that should accompany it. For example, if a user had just viewed metadata associated with an Assistment the PortalLogger would record an ASSISTMENTBROSWER_VIEW_DETAILS action type and the action details would contain the object id for that Assistment. Each action is associated to a user via the user id.

4.4.12 Miscellaneous Functionality

In addition to the tools created for the Assistment Portal, a PreferenceEngine was created which allows for storing and retrieving of preferences for a specific application or tool. This engine is globally accessible by all aspects of the Assistment Project. It is the responsibility of each application or tool to define its preferences and the meaning of each preference. The PreferenceEngine stores preferences in a database associated with its application or tool. Each preference has a set of allowable values while each user will have a value mapped to each preference they have set. Applications and tools can also specify a default value for a preference to be used if the user has not specified one. The engine’s only function is to store and retrieve preferences.

Since the Assistment Project is web based it is possible for a user to manually type the URL for an Assistment Portal tool in an attempt to use the application. This is a security issue in that any user has the ability to access tools not meant for them. To combat this problem each tool detects the user attempting to access the tool and verifies the user has permission to view this particular tool. If a user does not have the required permissions they are sent back to their main page.

5.0 Evaluation

In order to evaluate the success of the Assistment Portal redesign, an overarching goal of the Assistment Project was used: developing a solution to reduce the total cost of maintenance of an Intelligent Tutoring System. In this situation “maintenance” means the developing and deploying of content as well as the management of the groups and users in the system. As such the time that is required to construct and assign a curriculum was assessed. Additionally the time required to construct Assistments will be factored in to the time to create and deploy a curriculum in order to produce a total estimate for creating and deploying content via the Assistment Project.

The time of curriculum construction and assigning is calculated based on the actions recorded by the PortalLogger. The construction of a curriculum and the assigning of a curriculum were evaluated independently as not all built curriculums were assigned immediately or at all. Additionally the creation of manual curriculums is not included in this analysis as the analysis is aimed at evaluating the user-oriented tools created for that task.

Looking first at curriculum construction time, (see Table 1), users created curriculums with the Curriculum Tools and their times were recorded. A total of eleven users created thirty-five curriculums. Construction time ranged from approximately 14 seconds to approximately 414 seconds (6.9 minutes) with the average time being about 92 seconds (1.5 minutes).

User ID	Start Time	Elapsed Time (seconds)	Assistments per Curriculum
1	2005-11-28 09:21:14.00882	16.994	2
1	2005-11-28 09:38:34.00016	14.009	1
2	2005-11-28 09:16:09.00002	18.009	3
2	2005-11-28 09:33:55.00956	23.996	2
14	2005-11-15 15:02:54.00242	366.007	48
20	2005-11-22 16:33:48.00112	42.001	8
20	2005-11-22 16:47:13.00446	48.002	6
20	2005-11-29 13:17:35.00987	216.996	7
1161	2005-11-16 14:31:47.00506	177.997	10
4980	2005-11-16 12:00:14.0062	83.002	13
4980	2005-11-29 15:39:12.00843	22.999	12
5880	2005-11-17 15:28:57.00189	93.004	9
5880	2005-11-17 15:30:30.00536	56.998	9
5880	2005-11-30 00:14:41.00687	24.002	12
5880	2005-11-30 11:09:28.00725	32.998	15
6380	2005-11-29 11:47:35.00517	39.003	2
6380	2005-11-29 11:48:14.00725	93	7
6380	2005-11-29 12:35:40.00346	89.004	2
6380	2005-11-29 12:37:09.00718	71.997	2
6380	2005-11-29 12:38:21.00493	49.997	2
6380	2005-11-29 12:39:11.00122	37.999	1
6380	2005-11-29 14:44:44.0092	43.997	7
6628	2005-11-15 20:17:13.00433	270.004	1
6628	2005-11-16 12:34:58.00798	35.001	1
6628	2005-11-20 00:32:35.00539	414.998	25
7198	2005-11-15 13:07:46.00879	90.001	14
7198	2005-11-29 22:37:02.0073	101.996	22
7198	2005-11-30 08:42:35.00318	40.004	22
7198	2005-11-30 08:45:46.00464	99	12
17172	2005-11-28 11:56:18.00996	311.996	24
17172	2005-11-28 12:01:30.0058	90	24
17172	2005-11-30 22:57:05.00358	38	72
17172	2005-12-01 13:54:04.0059	27.996	72
17172	2005-12-02 20:43:17.00774	20.002	72
17172	2005-12-02 20:45:57.00981	28.993	99

Table 1. Curriculum Creation Time

Moving to the evaluation of assigning a curriculum (see Table 2), some individuals who created curriculums or had curriculums shared with them assigned them to classes; the time spent in this process was recorded. Seventeen users assigned curriculums for thirty-seven assignments. The shortest assignment time was approximately 11 seconds; the longest assignment time was approximately 853 seconds (14 minutes) with an average time of approximately 63 seconds (just over 1 minute). It is worth noting that the 14 minute assignment is the result of the user pausing in the middle of the assignment process to turn his attention to other matters and then, after some time, returning to complete the assignment; as a result this value is significantly higher than the next highest of approximately 134 seconds (2.2 minutes). If we leave out this extreme value we get an average of approximately 41 seconds which is a much better estimate of the time to assign a curriculum.

User ID	Start Time	Elapsed Time (seconds)
14	2005-11-15 15:31:37.00807	49.997
106	2005-11-29 13:56:13.00331	26.006
6628	2005-11-20 00:50:29.00981	134.996
7116	2005-11-08 14:52:28.00799	88.997
7198	2005-11-29 14:56:17.00136	30.005
7198	2005-11-29 15:00:42.00013	18.006
7198	2005-11-29 23:56:30.00813	12.001
7198	2005-11-30 00:04:05.00245	19.004
7198	2005-11-30 08:43:24.00827	27.997
7198	2005-11-30 08:47:28.00027	11.007
17172	2005-11-08 14:14:44.00797	80.999
17172	2005-11-30 23:39:39.00437	38.001
17172	2005-12-01 22:29:06.00584	853.003
17172	2005-12-02 20:10:14.00635	24.998
17172	2005-12-02 20:44:56.00154	26
17172	2005-12-02 20:47:33.0066	27.997
18317	2005-11-28 11:05:29.0074	30.998
19958	2005-11-28 13:14:44.00969	33.996
20111	2005-12-02 09:09:02.001	76.005
20111	2005-12-02 12:56:17.00135	48
20111	2005-12-02 12:57:11.0063	42.001
20116	2005-12-02 07:18:51.00592	45.001
20116	2005-12-02 07:21:34.00314	128.005
21590	2005-11-09 16:46:09.0073	9
21634	2005-11-28 11:45:42.00617	40.996
23564	2005-11-09 16:42:24.00709	25.994
24278	2005-11-14 08:14:38.00809	26.999
24586	2005-12-02 07:28:26.00697	25.998
24978	2005-11-29 15:03:42.00264	15.002
24978	2005-11-29 15:45:24.0072	12.002
24978	2005-11-29 22:32:00.00231	33.999
24978	2005-12-01 02:30:20.00053	55.003
24978	2005-12-01 02:34:50.00623	34

25021	2005-11-30 09:36:23.0062	52.999
25021	2005-11-30 09:40:40.00749	16.001
25021	2005-11-30 10:14:19.0089	23.995
25021	2005-11-30 12:06:57.00898	92.992

Table 2. Curriculum Assignment Time

Time to create Assistments via the Assistment Builder has been the subject of other research and was previously reported [14, 15]. The most recent numbers are based on the creation of over 270 Assistments. These Assistments varied in their complexity ranging from simple top-level only Assistments to multi-scaffold Assistments. The data collected indicates that it requires an average 20.68 minutes to build an Assistment with about 5.15 minutes per scaffold. Each Assistment provides approximately two minutes of content. It is important to note that the time recorded for creating an Assistment is an underestimate due to offline preparation. Such preparation consists of image construction and layout planning. Upon surveying users it was found that the estimate per Assistment should be raised to about 60 minutes.

6.0 Discussion

Attempting to compare the curriculum creation times to the previous amount of time it took to create a curriculum is difficult in that it was previously done manually and was not timed. To recap from earlier, the manual procedure for creating a curriculum was:

1. Locate all the Assistment IDs that are to be included in the curriculum
2. Construct an XML document for the style of curriculum being created using the Assistment IDs.
3. Manually insert the curriculum XML into a database.
4. Verify the curriculum XML is well-formed.

It is fairly easy to see that the previous process was quite involved, time intensive, slow, and in many cases required multiple individuals to complete the process, one of which was typically a senior Assistment Project team member with knowledge of the XML representation. It is easy to envision that this process would certainly take longer than an average of 1.5 minutes or even longer than the newly identified maximum of 6.9 minutes. In some cases curriculum construction took days to complete. By comparison, the new curriculum creation system is far less complicated and time consuming; thus the solution has been deemed a success. The new tool created for creating a curriculum requires no manual steps and does not require an Assistment Project team member to complete. As a result, the tools created have succeeded in creating a process that is usable by a user at reasonable speeds.

Likewise, attempting to compare the new estimates of the time required to assign a curriculum with previous times, is not possible as previous assigning of curriculums was a manual un-timed process. To recap from earlier, the manual procedure for assigning a curriculum was:

1. Locate the curriculum IDs for the curriculums to be assigned.
2. Locate the class IDs for the class that the curriculums will be assigned to.

3. Add a mapping to in a database to indicate the assignment.
4. Test to ensure the assignment is correct.

Much like curriculum creation it is easy to see that this process was drawn out and time consuming. Previously assigning a curriculum could take multiple minutes and required a senior Assistment Project developer to complete the process. In the case that no such developer was available the assignment would have to wait until one became available, further extending the time to create the assignment. Since the current process takes about 41 seconds, the assignment can be done by a novice user, and the speed of assigning a curriculum is reasonable the solution provided was deemed acceptable.

In terms of the extensibility and modularity of the developed architecture, recent additions and modifications to the system demonstrate the success of the approach. In terms of extensibility the main focus was on allowing extension of Assistment types and curriculum types. Recently a new Assistment type has been introduced: the range question. The range question allows for an answer to be considered correct if it falls within a specified range; this type of question is useful in tutoring a students estimation skills. For the addition of this new Assistment type, no code was changed in the CTOP or any other applications. The only change came in the addition of two classes to the CTOP. The two additions are a class representing the question type, including how to display it, and a class for evaluating the actions taken on the new question type. The API provided by CTOP can also be viewed as positive by the fact that applications have been built that use the API including the Assistment Portal. The API has provided a means to access the core objects of the system without duplicating code and/or modifying existing code in an attempt to access the data. As previously mentioned the goal was not to create a complete full-featured framework but instead to provide some services and features similar to a full-featured framework. Given the ability to extend the existing objects and the use of the provided API it is the CTOP solution does provide support for extensibility.

With regard to modularity recent work has provided an example of the modularity of the code base [10]. Towards the end of the work involving the creation of the DataLayer many issues became obvious with respect to system scalability under the Hibernate solution that was in use. Due to these performance concerns and a quickly approaching deadline, it was deemed necessary to remove the Hibernate DataLayer and write a custom DataLayer. This move was the driving force for a move to storing XML in the database. Given the time constrains on the redevelopment it was the quickest way to get a working DataLayer in place, while leaving open the possibility to migrate to a relational database. The new custom DataLayer was completed in time and plugged into the system in place of the Hibernate DataLayer with no need to change any other code in the system. The ability shown in this example, to swap out the DataLayer with no effect on any of the other code, speaks volumes on the success of the encapsulated modularity of the system.

The security requirements for the Assistment Portal were to establish a policy for access control to the tools. This goal was successfully met and can be evaluated programmatically. Each tool verifies the user attempting to access the tool based on their user type. If the user type is allowed they can proceed, if not the user is redirected back to their respective main view. The only way for a user to access a restricted part of the Assistment Portal is if the users account type was changed or if they obtained login information for an account type that is able to view the tool. The first situation would require direct access to the Assistment Project database and knowledge of the data model. The second situation, obtaining other users login information, is beyond the control of the system.

As mentioned earlier, it has been estimated that 200 hours of time was required to produce one hour of Intelligent Tutoring System content [1, 9]. The work for this thesis was an attempt to devise a system of tools to help lower this 200:1 ratio. Given that it has been shown that an Assistment produces 2 minutes of content, it would take 30 Assistments to produce one hour of content. Taking the estimate of 60 minutes of building time per Assistment, and 30 Assistments for one hour of content, we see that it will require 1800 minutes (or 30 hours) to produce one hour of content. In addition to the time required to build the Assistments, we need to factor in curriculum creation and deployment times. If we were to bundle the 30 Assistments into a curriculum it would on average require less than approximately 6.9 minutes. The assignment time of the curriculum is not a clear measurement, since it depends on how many assignments of the curriculum should be taken into account. Given that the developed curriculum assignment tool allows the user to assign the curriculum to multiple classes in one assignment, assigning to one class versus all the users classes can be treated as the same. Taking into account the adjusted maximum time for curriculum assignment we now have 1800 minutes to create 30 Assistments, 6.9 minutes to create a curriculum, and 2.2 minutes to assign a curriculum to classes; leaving a total time of 1809.1 minutes (30.15 hours) of development time for one hour of content. This gives a ratio of approximately 30:1 which is an improvement over the 200:1 ratio previously reported. Interestingly, the Assistments created under this system have been shown to produce meaningful learning among students [14].

It is important to notice that the development time of the curriculum and its assignment does not greatly impact the total time required to produce an hour of content. Perhaps if more advanced curriculums were to be developed they would weigh more heavily on the total time of construction and deployment. The comparison of the 30:1 ratio developed here and the 200:1 ratio previously reported is considered to be comparable even given that the 30:1 relates to the construction and deployment of pseudo-tutors while 200:1 relates to the construction and deployment of full rule based tutors, since pseudo-tutors have been shown to be behaviorally equivalent to rule-based tutors [6].

7.0 Future Work

The Assistment Portal will continue to evolve to meet the growing demands of its users. The user interfaces could be redesigned in a more user friendly way. A user interface study is needed to identify the best manner in which to present each tool and its associated data to the user.

The existing curriculum tools only support three basic types of curriculum construction. It is not possible to create complex curriculums within curriculums nor is possible to construct more advanced curriculums supported by the system via the tools. There is a system administrator level tool that is part of the Curriculum Tools which allows the direct editing of the underlying XML representation of a curriculum, which does allow for the creation of any type of curriculum. However, this type of editor is not usable by teachers and other individuals not familiar with our XML representation.

Similar to the *system administrator*'s direct XML manipulation tool for curriculums, there is a tool for directly manipulating Assistments. The intended purpose of this tool was for system administrators to repair malformed XML. This tool could be enhanced to provide more functionality to assist in the repair of corrupt Assistments.

The current permission schema for Assistments, curriculums, and classes is also very basic. It would be worth looking into creating a more robust permission system which includes update types to limit how a user can modify/use content. Also with regard to changing content, having built-in "version" control would allow for an easy undo in the event of erroneous deleting, corruption, or other damaging behaviors. This "versioning" behavior would allow users to "rollback" their content to a previously desired state.

8.0 Conclusion

The redesign and expansion of the CTOP, DataLayer, and Assistment Portal has been a successful undertaking. The results from the analysis of the solutions developed indicate users are able to create and deploy content with the new system and development and assigning speed of has been reduced. Additionally an overall time for development and deployment of content for the Assistment Project has been identified and is an improvement over previously reported time. The extensibility and modularity requirements have been incorporated within the new design and provide a foundation for which to build additional services and features upon. Moving forward more work can be done to alleviate user interface concerns, as well as the development of more advance tools. If new tools are needed they can easily be integrated into the Assistment framework, were previously much work was needed to add such services and features.

Reference

1. Anderson, J.R., (1993). *Rules of the Mind*. Hillsdale, NJ: Erlbaum.
2. Crowley RS, Legowski E, Medvedeva O, Tseytlin E, Roh E, Jukic DM. (2005). An ITS for medical classification problem-solving: Effects of tutoring and representations. In C.K. Looi, G. McCalla, B. Bredeweg, & J. Breuker (Eds.) *Proceedings of the 12th Artificial Intelligence In Education*, 555-562. Amsterdam: ISO Press.
3. Feng, M., Heffernan, N.T, Koedinger, K.R. *Addressing the Testing Challenge with a Web-Based E-Assessment System that Tutors as it Assesses*. Submitted to WWW2006, Edinburgh, Scotland (2006).
4. Heineman, G.T., Councill, W.T. (2001), *Component Based Software Engineering: Putting the pieces Together*, Boston, MA: Addison-Wesley Professional.
5. Hibernate Relational Mapping Tool. (2005). http://www.hibernate.org/
6. 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. pg.162-173.

7. 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.
8. Morgan, P., & Ritter, S. (2002). An experimental study of the effects of Cognitive Tutor® Algebra I on student knowledge and attitude. (Available from Carnegie Learning, Inc., 1200 Penn Avenue, Suite 150, Pittsburgh, PA 15222).
9. Murray, T. (1999). Authoring Intelligent Tutoring Systems: An Analysis of the State of the Art. *International Journal of Artificial Intelligence in Education*, 8, 30-43.
10. Nuzzo-Jones G., Macasek MA., Walonoski, JA., Rasmussen K., Heffernan NT. *Common Tutor Object Platform – an e-Learning Software Development Strategy.* Submitted to WWW2006, Edinburgh, Scotland (2006).
11. 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, & J. Breuker (Eds.) *Proceedings of the 12th Artificial Intelligence In Education*, 555-562. Amsterdam: ISO Press.
12. Published by Carnegie Mellon University (2003), Open Learning Initiative (OLI), http://www.cmu.edu/oli/.
13. Published by Massachusetts Department of Education (2005), Massachusetts Comprehensive Assessment System, http://www.doe.mass.edu/mcas/.
14. Razzaq, L, Feng, M., Nuzzo-Jones, G., Heffernan, N.T. et. al (2005). The Assistment Project: Blending Assessment and Assisting. In C.K. Looi, G. McCalla, B. Bredeweg, & J. Breuker (Eds.) *Proceedings of the 12th Artificial Intelligence In Education*, 555-562. Amsterdam: ISO Press.
15. Turner T.E., Lourenco A.L.N., Heffernan N.T., Macasek M.A., Nuzzo-Jones G., The Assistment Builder: An Analysis of ITS Content Creation Lifecycle, *Submitted to FLAIRS2006*, Florida, USA (2006).
16. 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, & J. Breuker (Eds.) *Proceedings of the 12th Artificial Intelligence In Education*, 555-562. Amsterdam: ISO Press.
17. Vendlinski, T., Niemi, D., Wang, J., Monempour, S., Lee, J. (2005). Improving Formative Assessment Practice with Educational Information Technology. *American Educational Research Association 2005 Annual Meeting*.
18. Walonoski, J., *Visual Feedback for Gaming Prevention in Intelligent Tutoring Systems*, Worcester Polytechnic Institute Master Thesis (2005).

Acknowledgements

I would like to thank the Worcester, MA area schools for their continuing support of the Assistment Project. Additionally I would like to acknowledge the US Department of Education, National Science Foundation, and Office of Naval Research, for providing the funding to accomplish this research.