Scaling up a Server-Based Web Tutor: A Description of System’s Scalability & Reliability Efforts

Jozsef PATVARCZKI, Neil T. HEFFERNAN & Kai RASMUSSEN
Worcester Polytechnic Institute
{patvarcz,nth}@WPI.EDU

Abstract. Web-Based Intelligent Tutors offer many benefits. Because they are easily accessible, they help lower the entry barrier for teachers and enable both teachers and researchers to collect data and generate reports. Web-Based Intelligent Tutors usually run on a central server. By running the software for the Intelligent Tutor on a central server it’s possible to easily distribute content, deploy new features, and allow researchers to more easily run randomized controlled studies. Two concerns when running the Intelligent Tutor on a central server are: 1) building a scalable server architecture; 2) providing reliable service to researchers, teachers, and students. Our research team has built a web-based tutor, located at www.ASSISTment.org, that is used by hundreds of students a day in Worcester and surrounding towns. In this paper we will answer several research questions: 1) Can we reduce the cost of authoring ITS; 2) how can we improve performance and reliability with a better server architecture. We will also describe our architecture and conclude with some recommendations for new Intelligent Tutoring System builders.

Introduction

Its cliché now to say, but the Internet changes everything. While Intelligent Tutoring Systems have been around for over 30 years, the simple fact of the Internet changes the way ITS can be built. These changes include how the system’s content should be built (teachers can be made to build them), how they are to be deployed (internet web-sites versus application that run on desktops), how data is collected and analyzed for teacher reports, and how the system can be used by scientists to study learning through randomized controlled experiments.

This paper describes aspects of the ASSISTment System that relate to it being a sever-based architecture (as opposed to a stand along application); in particular the relation to its scalability, reliability and extensibility. Within the last year, the ASSISTment system has grown significantly. The system is now being used by 3000 students, which is almost a two-fold increase. They system now has twice as many teacher and schools. This growth has caused growing pains which we will have been seeking solutions to. For results on how much students learn from the ASSISTment see [5]. For results related to how well a job we do assessing students see [1].

We begin by looking at our first research question to see if we can build a scalable system that will allow us to reduce the cost of authoring content.
1. Scalability Related Work

Intelligent tutors have been shown to be effective at teaching grade-school math to children [1]. With the notable exception of Carnegie Learning’s Cognitive Algebra Tutor [6], there are no commercially successful Intelligent Tutoring Systems (ITS). Tom Murray, Stephen Blessing, and Sharon Ainsworth's recent book [5] reviewed 28 authoring system for ITSs. Unfortunately, they found that there is not a single system that is of "release quality", let alone commercially available. Murray ([5], p.491) asked why there are not more ITSs and proposed that a major part of the problem was that there were few useful tools to support ITS creation. WPI and CMU have been funded by the Office of Naval Research (which funded much of the CMU effort to build the algebra tutor) to explore ways to reduce the cost associated with creating cognitive model-based tutors used in ITSs [7] [3]. Traditional ITSs [4] [1] have been built by programmers who need PhD-level experience in AI rule-based programming as well as a background in cognitive psychology.

Murray, after looking at many authoring tools [2] said “A very rough estimate of 300 hours of development time per hour of on-line instruction is commonly used for the development time of traditional CAI.” While building intelligent tutors systems is generally agreed to be much harder Anderson [1] suggested it took at least 200:1 to build the Cognitive Tutor at CMU so we choose this as our comparison ratio.

![Figure 1: The builder and associated student screen](image-url)

We are taking a different approach than supporting teachers learning how to do AI-rule-based programming. We hope to lower the skills needed to the point that normal
classroom teachers can author their own ITS content. Our approach is to use "example tracing" tutors [6] to reduce the amount of expertise and time it takes to create an intelligent tutor, thus reducing the cost. The goal is to allow both educators and researchers to create tutors without even basic knowledge of how to program a computer. Towards this end, we have developed the ASSISTments ITS; a web-based tutor authoring, administering, and reporting system.

So while the previous literature has suggested that there is a cost of is at least 200 hours of building intelligent tutors for each hour of instruction, our goal is build a system that will both reduce this authoring ratio, as well as reduce the skills needed by authors to use the system.

1.1 Scalability of Creating Web-Based Portals for Authoring

Users begin interacting with our system through the “Portal” that manages all activities in the system. Teacher get accounts that allow them to do many things including managing their class rosters, getting reports on student, and creating new content, using the Builder, for their classes. Figure 1a shows a few of the screens teachers use to do these tasks. See [1,2] for more details as well as [5] for a more complete description of the build. [5] also reported initial results on the average time it took to create an ASSISTment. Figure 1 shows an example of the Builder as well as the a screen shot of what the student sees.

![Figure 1: A diagram showing screen shots of the different things teachers can do.](image-url)
1.1.1 Experiment of the Scalability of Creating Content

One of the components of the ASSISTment system is a web-based content creation tool called the Builder, which is shown in left hand portion of Figure 1. The Builder’s interface uses common web technologies such as HTML and JavaScript, allowing it to be used on most modern browsers. The Builder allows a user to create pseudo tutors composed of an original question and scaffolding questions which break down the original question into different steps. In the next section, we evaluate this approach in terms of usability and decreased creation time of tutors.

1.2.1 Experiment Methodology

We wished to create a new 10th grade math tutoring system to complement our existing 8th grade math system. In September 2006, a group of nine WPI undergrads began to create 10th grade math content as part of an undergraduate project focused on relating science and technology to society. Their goal was to create as much content as possible for this system. All content was first approved by the project’s subject-matter expert, an experienced math teacher. We augmented our builder tool to track how long it takes students to create an ASSISTment. This does ignore the time it takes student to plan the ASSISTment, work with their subject-matter expert, and any time spent making images and animated gifs. All of this time can be substantial, so we can’t claim to have tracked all time associated with creating content. Once we know how many ASSISTments they have created, we can estimate the amount of content tutoring time created by using the previously established number that students spend about 2 minutes per ASSISTment. This number is averaged from data from thousands of students. This will give us a ratio that we can compare against the literature suggesting a 200:1 ratio. [CITE] We also gave students a 1 hour tutorial on the tool.

1.3.1 Experiment Results

The nine students created 121 ASSISTments in the first 7 weeks of the project with no ASSISTance from the ASSISTment staff other than meeting with their subject matter expert to review the pedagogy. We know from prior studies ([Heffernan, 2006]) that students on average spent 2 minutes per ASSISTment, so the IQP students created 242 minutes, or a little over 4 hours of content. The log files were analyzed to determine that students spent 79 minutes (std = 30 minutes) on average to create an ASSISTment. In the second 7 weeks, they created 115 more additional ASSISTments at a rate of 55 minutes per ASSISTment. This increased rate of creation was statistically significant (p<.01), suggesting that students were getting faster at creating content. To look for other learning curves, we noticed that in the first 7 week, each ASSISTment was edited on average over the space of 4 days, (std= 1.2 days) while in second time weeks, the IQP students were only editing an ASSISTment 3 times on average. This rate that was statistically significantly faster than in the first 7-week period.
1.4.1 Experiment Discussion

It appears that we have created a method for creating intelligent tutoring content much more cost effectively. We did this by building a tool that reduces both the skills needed to create content as well as the time needed to do so. This reduces a 200:1 ratio to a ratio of about 80:2.

2. Web-Performance scalability

We now present our old architecture as well as our steps towards making a move towards a scalable and fault tolerant architecture. One year ago our system was a single TOMCAT application server connected to a single database server, as shown in Figure 3. Our old architecture is illustrated by dark outlined boxes. This architecture had multiple single points of failure. Our current architecture, also shown in Figure 3, incorporated scalability at our first entry point through the use of a virtual IP for www.assistment.org, provided by the CARP protocol, and a load balancer to distribute load over multiple TOMCAT application servers. Finally, we plan to address scalability in our database design. We currently have two TOMCAT servers with plans to add more. Below we will report results on our increased performance using multiple TOMCATs, which would allow a greater number of maximum simultaneous users able to use this new system.

In addition to scalability, we have to be concerned about reliability. Unfortunately, in the past teachers calling Professor Heffernan’s cell phone to report problems was our fault-detection system. Our new monitoring system, which uses Selenium, has allowed us to send text messages to our administrators when the system goes down.

![Figure 3: Current and new architecture](image.png)

2.1 Evaluation of Real Student Respond Data

Our first research question was to investigate the average time it takes our server to create the next web-page to send to a student. We wished to see how that average time under
different loads (i.e. number of students simultaneously using our system). We added additional logging information to record the time it took to generate the page and the number of HTML responses. We collected two and half months work of real student data and aggregated the data into 10 minutes intervals. We then calculated an average page creation time for each 10 minute interval. We then plotted in Figure 3 the average page creation time for these intervals against the number of simulation unique users.

2.1.1 The Results and Discussion

The most important result shown in Figure 4 is that all but a handful of these 10-minute interval have average page creation times that are less then 1 second. Most of these are clustered around 100 milliseconds.

Since each public school classes have about 20 students, we noticed clusters (shown in ovals in the bottom left) of intervals where a single class was logged on. We noticed a second cluster of around 40 users, which most likely represents instances where two classes of students were using the system simultaneously.

Surprising to us, even the intervals where 80-100 users were on the system simultaneously, there was no appreciable pattern towards a slower page creation time. This was true, even when we had 80-100 users on the system and we were generating over 1000 pages per interval (100 pages per minute). Of course, with more users, we expect to see a slow down, but we have not yet reached that point. Additionally, there are a few intervals with an average page creation time greater then 1 second (top-left oval). We inspected our
logs, and found out those intervals were associated with times when an entire class of students would log on simultaneously. The log-on procedures is the most expensive step in the process and this data shows that this might be a good place for us to improve.

2.2 Stress Testing with Simulated Students

We wanted to evaluate our decision to increase scalability by adding TOMCAT application server to our load-balanced infrastructure. Our research question was whether we could get a linear speed-up with additional Tomcat servers or not. We also wanted to see if a caching method could increase performance. In our methodology we proposed to measure the speed up by simulating students using the Apache JMeter framework. We simulated three different scenarios. In the first scenario we used 50 threads simulating 50 students working with our old infrastructure (no load-balancer, one tomcat box, and one database server). We used an average 10 seconds random delay between each student action that represents a typical student reaction time. In our second scenario we changed the infrastructure and we used the load-balancer to distribute the load between two servers (load balancer, two Tomcat boxes, and one database server). Finally in our third scenario, we used a web-cache technique without load-balancer and one Tomcat box with a single database.

2.2.1 The Results and Discussion

During the execution of the tests the memory and CPU utilizations were normal at the Tomcat and at the database side as well. For the test, we used a Pentium 4 with 3GHz CPU and 2GB RAM. The table 1 represents the results for the test scenarios.

<table>
<thead>
<tr>
<th>Test Type</th>
<th>Number of Unique Users</th>
<th>Response Time [ms]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scenario 1/Normal</td>
<td>50</td>
<td>8955</td>
</tr>
<tr>
<td>Scenario 2/LB</td>
<td>50</td>
<td>3624</td>
</tr>
<tr>
<td>Scenario 3/Web-cache</td>
<td>50</td>
<td>8073</td>
</tr>
</tbody>
</table>

Table 1: Test results showing the Load-balancer cut the average response time in half

According to the results of the first scenario we got 9 seconds for average response time, which looked to high. During the test we “played student” and found that the average response time was much less than 9 seconds, which led us to conclude that JMeter was failing to stress the systems quickly enough. Nevertheless, it does seem that the load balancer did appear to cut the response time in half, however the caching method had a much less significant role in reducing the average response time.

We seem to have able to get linear speed-up by the help of the load-balancer and an additional Tomcat server.

2.4 Scaling up with Extensibility and Collaboration
Intelligent Tutoring Systems are customarily thought of as closed systems. With such a strong research community, it would seem beneficial for researchers to be able to work together; using contents and tutoring methods from within other existing system. With this collaboration, existing tutors could reach broader audience. The question remains: how do two seemingly different tutors cohabitate peacefully? We have solved this question by using web services.

A web service is an interoperable piece of computer logic accessible through standard Internet protocols and powered by XML communication. Because data is represented with XML and communications between applications are achieved with standard protocols, web services remains platform and technology independent. Because of this, two disparate tutoring systems can interact easily using this technology. Researchers could include content and currently unsupported tutor types from other systems. Web services can enable a more collaborative and extensible system.

We have extended the ASSISTment system to be able to perform evaluation of student input through web services. This web service is described by an XML interface which can be implemented in a variety of different ways and in a variety of different languages. We currently have ASSISTment content hosted as a web service. Because of this, by implementing a web service interface, another ITS could incorporate ASSISTment content into their system. The ASSISTment project is looking to collaborate with other systems such as the TeachScheme project to include a Scheme programming language tutor. This is just one example of how web services have enabled the ASSISTment project to become more collaborative and extensible.

3. Conclusion

We have reported some positive results with regard to trying to scale up ITS to make it less costly to build ITS. In our system non-programmers can create ITS content relatively quickly compared to the times reported in the literate to create intelligent tutoring systems. We also have found some positive results in our attempt to create an architecture that will scale-up better. We hope that this paper helps other ITS create build more scalable systems.

References