Computer Science Education
1998, Vol. 8, No. 3, pp. 251–264
0899-3408/98/0803–0251$12.00
© Swets & Zeitlinger

Poser, an Online Review Tool in Java

Mark Lattanzi and Dan Cosley
James Madison University, Harrisonburg

ABSTRACT

This paper presents an online reviewing tool, Poser, written in the Java programming language. Online quizzing tools have proliferated lately, but most are not so well-suited for reviewing. Poser's basic functionality is to provide students with a convenient, interactive way to review course materials in a question/answer format. Incorrect responses are corrected, along with an optional explanation. The program keeps usage statistics and information on responses to questions. Poser can be an important part of any course, because it gives students an easy way to test their understanding of the course materials from any Internet computer. We used Poser in two introductory Computer Science courses and gathered some formative data which was used to modify and evaluate the program. Over 100 students used the program and they answered over 38,000 questions while reviewing for their final exams. Student comments indicate that the program was very helpful. Overall, we feel that Poser was a huge success and are planning a controlled study in the coming semesters to measure how much the use of the program facilitates student learning and understanding.

INTRODUCTION

Poser is an online program that students can use to review course material from any web browser. Unlike the many quizzing programs already on the net, this one is different in that it is a review tool rather than a quiz program. Poser uses a Java applet and server to ask questions of the user. When a question is answered, the program provides feedback (either congratulations or the correct answer with an explanation). Poser then displays another question to be answered.

We have placed the Poser applet on the home pages for several classes in the department. When students come to the page to get notes, check grades,

Correspondence: Dr. Mark Lattanzi, 209 ISAT/CS Bldg, James Madison University, Harrisonburg, VA 22807, USA. Tel: 540-568-2777. E-mail: lattanmr@jmu.edu

etc., they often take the time to answer a question or two—and, near test time, a question or twenty. This is an easy way for instructors to reinforce the class material. As an added bonus, by controlling which questions Poser asks, instructors can emphasize the material they consider most important. Finally, Poser gathers statistics on which questions are asked and the answers provided. This helps instructors determine where students may need more assistance. It also helps instructors identify the common misunderstandings that students may hold.

BACKGROUND

Academic Tools (Atools) Project

Poser is one of several programs developed at James Madison under the rubric of the Atools ("academic tools") project. Started in spring of 1997, the archive includes utilities designed to make administering classes less burdensome so that professors can spend more time on course content and quality. The collection contains several utilities, including a self-maintaining bulletin board suitable for locating tutoring services or exchanging textbooks, and a graphical mailing list manager, written in Java, that allows for easy mailing list maintenance over the web.

A natural candidate for addition to the archive was an online testing program. However, many people have already done online testing and, in fact, the web is replete with this kind of tool. The University of Hawaii's Ed Tech Tools center maintains a list of over 40 quiz-related programs, most of which are online (Shadian, 1998). Below are descriptions of a representative sample of the web-based quizzing software currently available.

Current Quiz Software

Most online quiz programs use CGI and HTML forms to administer quizzes. This is a natural extension of the form validation that has long been used with CGI scripts to verify information requests and product orders. The typical CGI quiz program uses a server-side program to compile quiz files into HTML. The quiz files themselves are usually text files with some sort of structure to indicate questions, ranging from simple delimiters to SGML-based extensions of HTML. Some programs include an editor that shields the user from the quiz file syntax. Either way, after the quiz file is generated, it is translated into standard HTML. Questions are constructed using form elements, making it easy to ask multiple-choice and true–false questions. Other question types, such as essay and short-answer, are also possible.

When students want to take a quiz, they go to the quiz URL. Many programs ask for a user name and password, both to record data and provide some measure of security. Once this information is entered, the quiz is presented. Some programs, including Mallard (Brown, 1998) and QuizPlease (McCormack, 1997), include ways to receive hints while taking the quiz; most require all the questions to be completed before doing anything else. Once students complete a quiz, they submit their answers. These are passed back to a CGI script on the server. This script typically grades the objective portion of the quiz, records results, and provides feedback to the student. Non-objective questions (e.g., short answers and essays) can be mailed to a grader.

Of course, there are variations on this theme. A few programs use Javascript. This allows for greater interactivity; however, if the grading is done on the client side, students can browse through the page's source code in an effort to find the answer. Two examples are JBC (Half-Baked Software, 1997) and the Javascript QuizMaker (Attotron Biosensor Corporation, 1998). A few Java-based programs, including JavaQuiz (Creagan, 1996), are also available. Some programs offered nice features like randomizing the order of questions and/or choices. Other good ideas included hints, explanations, and resources for students to research questions.

JUSTIFICATION

A Reviewer, Not a Quizzer

Clearly, the world does not need another online quizzing program. However, we decided that there is room for an online review tool that combines several desirable traits. Poser is:

- **Review-oriented.** Although any of these quiz programs *could* be used for reviewing, most are better suited for actual quizzes. Several factors contribute to this, including security measures, limits to the interactivity of HTML forms, and the common reaction of students to a "quiz". Poser's main theme is to increase student learning by providing a tool that allows students to test their current knowledge effectively.
- **Interactive.** We dislike the way most tools present an entire quiz on one web page. An eight-screen lump of quiz questions can be a disheartening experience for students. Poser asks one question at a time, provides immediate

feedback after each question, and allows students greater control over the program. These traits make Poser much more interactive (and less daunting) than other available tools. This results in increased student use and learning (hopefully).

- **Single-task.** One of the philosophies behind the Atools archive is that each tool should be simple and self-contained. Several of the better quiz tools available are parts of larger suites, including Mallard and WebCT (Goldberg et al., 1996). Though there is merit in large, powerful programs, they lead to additional complexity. If what you want is simply a review tool, a self-contained program like Poser is more appropriate. Poser is designed to be quickly and easily installed and configured for use.
- **Easy to use.** This trait is also born of the Atools philosophy of simplicity. QUIZIT, for example, is a powerful program that can track student progress and history, and can ask adaptive quizzes (Tinoco et al., 1996). However, instructors must learn a new syntax just to produce question files. It also requires someone to install and maintain mSQL (a UNIX database server) and to run programs to convert quiz files into HTML. Poser uses text files for its questions (optionally created by a Java graphical application), and requires no additional system tools for its use.

In addition, we wanted a program that was portable, flexible, and free. Some of the programs available run on only one platform (typically UNIX). Many were also limited to multiple-choice questions. By designing Poser in Java, these problems disappear. The program can run on any Java-enabled machine, and new question types can be easily added by extending two of Poser's classes. The Question class understands how to read in review questions from the text file, while the QuestionPanel class displays each question type in a nicely formatted way. In addition, since Poser is free (source code and executables), it is easier for professors to experiment and evaluate with it.

Advantages

The advantages of Poser are many.

- Poser helps students study.
- Poser is readily available.
- Poser is easy and fun to use.
- Poser provides student feedback to the instructor.
- Poser emphasizes important points.

Each of these is discussed in turn.

Student review
Poser helps students review the course material as the course progresses, instead of cramming just before an exam. Students regularly go to the course home page to retrieve notes from the week's lectures, get copies of class assignments, review their grades, check the message board, or follow newly posted links to pertinent websites. Each time a student visits the home page, Poser is right there. Many students feel compelled to answer a question or two before moving on. These little visits to the course material reinforce student learning.

Readily available
Poser is readily available from any machine with a web browser. Since Poser is written in Java 1.0, it can be viewed from any Java-enabled web browser without any extra installation hassles. Students can (and do) access the program from computer labs and from their networked dorm rooms. Accessibility is one key to getting students to use Poser; software that must be downloaded and installed by students may not be used to its fullest potential.

Easy to use
Another key to Poser's success is its ease of use. When the web page is loaded, Poser is waiting with a question or a list of topics and is ready to go. The questions are in the familiar true/false, fill-in, and multiple-choice formats and the interface for answering is simple—a push button, a textfield, a radio button. Upon answering, students receive feedback and then a new question.

Fun to use
Poser is fun to use. We have seen students compete for most questions answered correctly and they can also set up informal "Jeopardy/Trivial Pursuit" type games. Students also talk to each other about the various questions they have been asked (in effect, comparing notes).

Instructor feedback
Although Poser does not keep track of which students are answering questions, the server does keep summary data on questions asked and answers given. This data can be analyzed to see where students are having trouble. Extending the system to log individual student information would detract from the program's ease of use and popularity—but it would not be difficult, and Poser could then track individual students' needs or administer actual online quizzes.

Emphasis of selected points
Instructors can also use their control over the questions asked to emphasize important points from their lectures. As Erickson and Strommer point out, it is not always easy for students to identify the key ideas from a lecture or discussion (Erickson and Strommer, 1991). Careful selection of questions can help students focus their attention on the main issues.

POSER ARCHITECTURE

Poser is a relatively small and simple educational tool written entirely in Java 1.0 for portability. The software has three main components: a client applet, a server, and a utility for creating questions in an appropriate format. The actual question data reside in plain text files ("question files" or "quiz files") and complete the system.

Poser System Architecture

The interface to the user is a Java applet that can be easily included on any web page. Figure 1 shows the applet's interface. A question is displayed for the student to answer. After selecting a choice, the student clicks the "Submit Answer" button and receives immediate feedback. The applet keeps track of how many questions have been tried and answered correctly. The "Reset Stats" button resets these counters to zero. At any time, the student can select a different quiz (set of questions) to review with the "Select New Topic" button.

Poser can be included on any web page with the following code:

```
<APPLET    code=“poser.PoserApplet”    width=“100”
height=“40”>
<!— autostart is a mode where the applet automati-
cally starts a quiz when the page is loaded. The
default is no autostart —>
<PARAM name=“autostart” value=“HTML.quiz”>
</APPLET>
```

Complete instructions are included with Poser's documentation. Once included in a web page, the applet will work in any Java-enabled browser (Netscape or Internet Explorer, 3.x or higher).

The next piece of the system is a Java server, which acts as a backend to the applet. The server typically runs on the same host as the web server, but does

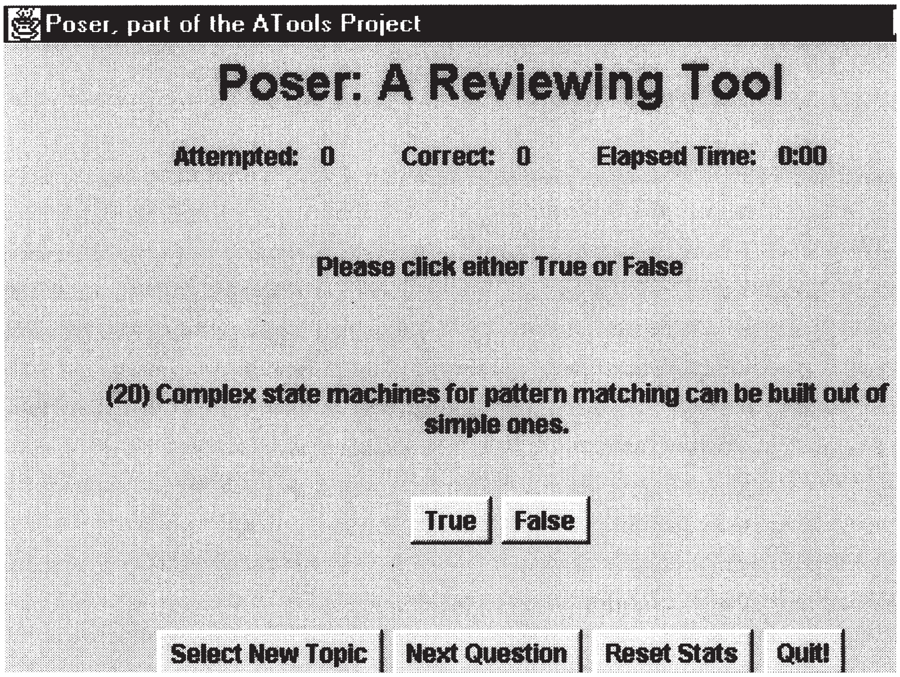
Poser, part of the ATools Project
Poser: A Reviewing Tool
Attempted: 0
Correct: 0
Elapsed Time: 0:00
Please click either True or False
(20) Complex state machines for pattern matching can be built out of simple ones.
True
False
Select New Topic
Next Question
Reset Stats
Quit!

Fig. 1. The Poser applet interface.

not have to. It reads in the data files of questions and quizzes, and listens for requests from Poser applets. If its data files are changed, it rereads them on the fly. More than one server can be run on the same machine. Figure 2 shows how the main parts of the system interact.

The "QuizMaker", also shown in Figure 2, is a Java application that instructors can use to create question files. These files are delimited text, so they can be edited directly; the quiz maker adds a convenient graphical interface for manipulating the files. Users can change the order of questions; add, delete, and duplicate questions; and edit questions with an interface tailored to the question type. These question files are then read in by the PoserServer (top middle of Fig. 2) and each question is parsed and stored by its type (multiple-choice, true–false, etc.). The PoserServer has been previously installed, configured with the text file quiz.ini, and run by the class instructor.

Whenever a student launches a PoserApplet (a simple click on the course web page), the server sends a question to the applet to ask the student. The QuestionPanel class in the applet displays the question in the student's browser

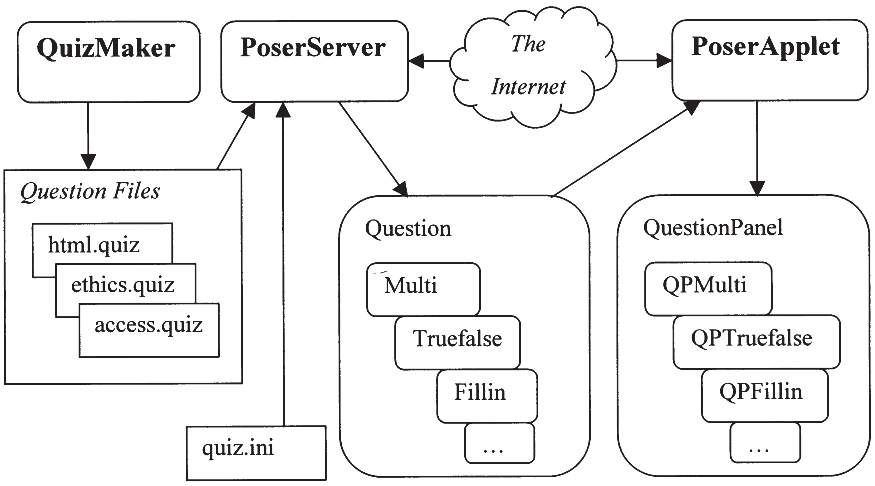

Fig. 2. The Poser architecture.

window. The student then selects (or types in) their answer. Upon receiving an answer, the applet evaluates it and provides feedback to the user. After reading the feedback, the user can request another question. The applet fetches the question from the server and the process starts again.

The server keeps track of which questions it has sent to each PoserApplet connected to it, so many students can use the software simultaneously.

Poser Database Files

Poser's database consists of delimited text files, including a collection of question files, an initialization file and a collection of question files. Figure 3 shows part of a question file. These files contain several questions, each of which has a type, the actual question (and distractors for multiple-choice), the correct answer, and an (optional) explanation. Currently, Poser "knows" three types of questions: multiple-choice, true/false, and fill-in-the-blank. The initialization file tells the server which question files to include and the name to be presented to users for each file. Additional topics can be added to the initialization file while the server is running

Experiment

Spring 1998 was the first semester that Poser was used. It was used by 55 students in an introductory computer literacy course (CS 138) to review for two

```
Multi
All of the following tags are required in a proper HTML
document except:
4
<HEAD></HEAD>
<TITLE></TITLE>
<BODY></BODY>
<HTML></HTML>
2

Truefalse
HTML is more oriented toward logical structure than phys-
ical layout.
T

Fillin
Additional information inside an HTML tag is called an __.
Attribute
```

Fig. 3. Example of part of a question file.

mid-term tests. These students were from all academic levels and represented many departments in the university. Based on their initial feedback, the program's interface and statistic gathering were improved.

It was then reinstalled with more questions, so the CS 138 students could use it to review for their final exam.

Another professor installed Poser for the final exam in an introductory programming course (CS 139). Fifty students in this course used the program.

Procedure

For the formative portion of this experiment, a set of question files on the CS 138 course material was created and set up so the students could review for their first two examinations. A file was created for each of the major topics in the course and added to the Poser server one to two weeks before the test. Use of the system was strictly optional, and students were encouraged to send feedback concerning their experiences with Poser. This feedback was then used to modify the Poser interface and to modify the Poser server to gather more data about the program's usage.

After revamping the program, it was reinstalled with a new set of question files for the CS 138 students to use to review for their final exam. In addition, a second server was installed for our CS 139 class to use in their review for their final. In CS 138, 204 different questions across 10 topics were created. The CS 139 students had 14 topics with 403 total questions. The Poser server recorded the total number of student sessions and the total number of questions asked in each session. Poser does not collect any statistics on individual student performance, and, although it does track the number of times each question is asked and the percentage of each answer given for each question, these data were not analyzed in this study.

RESULTS

Initially, we gathered subjective feedback and commentary on the installation and use of the program. Afterwards, we tabulated log files and examined the usage of the program objectively.

Student Perspective

Student comments were collected for the first two tests in CS 138. These comments were generally positive. Most said that the Poser helped them prepare for the tests. After the first test, suggestions to change the interface and to prevent questions in a question list from repeating were implemented. After the second test, several students suggested that the program give feedback explaining why the correct answer was correct.

Instructor Perspective

Setting up Poser and creating the data files is simple. Several graduate assistants and seniors were asked to create question files to use with the system and had no difficulty doing so. Question files and the server's initialization file can be changed on the fly. The log file can easily be processed to determine usage patterns (if the instructor desires). One instructor suggested a small tool to process Poser's log files. The tool could analyze these files (much as web statistics tools analyze web server log files), generate a simple report of Poser's usage, and automatically mail the report back to the instructor. This feature would allow instructors to easily monitor the usefulness of Poser (to their students) and is under consideration for the next revision.

Usage Frequency

A rough analysis of the logs from the first two tests for CS 138 supported the generally positive nature of the student comments mentioned above. About 10,000 questions were requested for the first test. This number was probably inflated, because of the repeating question problem mentioned above. It was surprising, then, to see that for the second test over 21,000 question requests were logged—after the change to eliminate repeating questions was implemented. Though tentative, this suggests that students who tried Poser on the first test found it useful and spread the word to their classmates.

Table 1 shows the usage realized by the Poser software across four examinations in the two different classes. Each of the columns represents the usage of the package for the two-week period right before the examination. The table contains some blanks because Poser was still under development during the first two tests in the CS 138 course.

The first row represents the total number of students that used the Poser software in each of the classes. Since Poser does not log student names (by design), a survey was conducted in each class to determine the total number of students that used the program. Row two shows how many questions were asked of these students during their studying/reviewing time. Data for rows three through six were only collected by the modified server for the two final examinations. Row three shows the total number of unique questions available for each examination. Row four shows the total number of different student sessions. The next two rows, five and six, show the minimum and maximum number of questions asked during a given session. Rows seven through nine are derived measures from the collected data showing the average questions per student and per session, and the average number of individual sessions performed by each student.

Analysis

Analyzing the data in Table 1, we find some interesting results. The CS 138 student usage increased across the three examinations. We believe this trend is the direct result of students realizing the benefits to be gained by using the reviewer. Student testimony to their classmates increased the popularity of the reviewer. The student comments that were solicited support this hypothesis. The CS 139 usage logs showed that many of the students found the program quite helpful (in that they answered 10 or more questions in a session), while about 15 percent of them answered less than 10 questions. Since the CS 139 students were seeing the program for the first time, this was expected. Had the

Table 1. Usage Statistics for the Poser Software.

	Column #	1	2	3	4
Row #	Group	CS 138 Test 1	CS 138 Test 2	CS 138 Final	CS 139 Final
1	Number of Students in Class	55	55	55	50
2	Total Questions Asked	10000	21000	24500	13500
3	Total Unique Questions	–	–	204	406
4	Number of Sessions	–	–	363	293
5	Minimum Questions / Session	–	–	1	1
6	Maximum Questions / Session	–	–	550	783
7	Questions / Student	182	382	445	270
8	Questions / Session	–	–	67.5	46
9	Sessions / Student	–	–	6.6	5.9

class been more familiar with the program, we believe that the total usage would have been greater.

Row seven warrants some explanation. Referring to the final exam for CS 138, if all 55 students using the program answered every question, then $55 \times 204 = 11{,}220$ questions would have been asked. So, since 24,000 questions were asked, on average each student went through the entire question database twice during their studying. Clearly, the students in CS 138 were using Poser to its fullest benefit by going through the questions multiple times, testing their current knowledge. In fact, row nine shows that, on average, each student in the class used the program on six different occasions.

Analyzing the final exam data for CS 139, we find that if all 50 students had answered all 406 questions, then 20,300 questions would have been asked. Here, we see that this was not the case. Only 13,500 questions were asked. We attribute this to the fact that this was the first time CS 139 students had been exposed to the program. Many students only answered one or two questions. However, even the students in CS 139 used the program almost six different times on average (row nine).

The above data clearly shows that Poser was a very popular program. We have no correlation between usage and grade achieved on the examination since Poser does not log student names. It would also be nice to see if using Poser actually increases student learning, but no data exists on past student performance (like standardized test scores). A formal study is being constructed from the Fall of 1998 to provide insights into both of these situations.

SUMMARY

Poser is still in its infancy. Tightly controlled empirical studies should be performed to validate its usefulness for student understanding and learning, but this preliminary investigation shows that the program is quite popular as a study aid and well-received by the student populations of both of the test classes. Across the entire first semester of the program's use, over 70,000 questions were asked to 105 students. Furthermore, the program was launched over 600 times by the students while studying for their final exams.

College students are sometimes stereotyped as not having great study skills and habits, but this experiment has shown that given a readily available, easy to use reviewing tool, students will use it extensively in their learning and studying of the course material.

FUTURE WORK

Poser is a newborn product that appears to have potential. We have plans for improving and developing it to be even more useful as a reviewing tool and as a tool to verify student learning and understanding. However, since much of Poser's virtue is its simplicity, we do not want to add too much functionality to the program, lest it become unwieldy for instructors to set up and maintain and too complex for students to want to use. One non-disruptive change would be to extend Poser to recognize new question types. Because of the object-oriented design of the program, these new types could be added without modifying the system architecture.

Another obvious addition would be to implement user logging and use this feature to track individual student difficulties and to collect grading information. This could be very useful information; however, we have several reservations. Adding a login mechanism and recording student performance would make Poser more complex. We are also afraid that some students would not use the program if they knew that their responses were being recorded. This would defeat the purpose of the tool.

Finally, although adding logins and recording would make actual quizzing possible, it is not obvious that doing this would be better or more convenient than the traditional quiz. Jeff Hawkins, the inventor of the PalmPilot personal organizer, described his company's competition, not against other products, but against pencil and paper (Gewirtz, 1998). Likewise, Poser's "competition" is not other computer software. Rather, it needs to be good, useful, and simple enough to break established patterns. Instructors must decide it is worth the effort to add Poser to their courses and students must find it to be a useful study aid.

To that end, some other enhancements in the works include:

- developing helper applications for analyzing the Poser's log files and for generating question files more efficiently;
- creating an online configuration tool for instructors as an alternative to having to use text configuration files on the server side; and
- presenting students with a summary page at the end of each session, showing the number of questions answered correctly and the total time spent.

A formal empirical study is being designed and will be implemented in the Fall 1998 semester in several classes. Poser can be found at http://atools.cs.jmu.edu/Poser/. It comes with all the necessary files, documentation on its installation and use, and a few sample quizzes on Computer Literacy. Poser is distributed free of charge.

REFERENCES

Attotron Biosensor Corporation (1998). Javascript QuizMaker Home Page [Online]. Available: http://www.attotron.com/pub/Quizmaker.html [1998, May 30].

Brown, D. (1998). Mallard Nestpage [Online]. Available: http://www.cen.uiuc.edu/Mallard/ [1998, May 30].

Creagan, D. (1996). JavaQuiz: A quiz shell in Java by Dan Creagan [Online]. Available: http://204.233.101.40/javaquiz/quiz.html [1998, May 30].

Erickson, B.L., & Strommer, D.W. (1991). *Teaching college freshmen*. San Francisco: Jossey-Bass Publishers.

Gewirtz, D. (1998, February). The PalmPower Interview: Jeff Hawkins, creator of the PalmPilot. PalmPower Magazine [Online], 42 paragraphs. Available: http://www.palmpower.com/issues/issue199802/hawkinstwo001.html [1998, May 30].

Goldberg, M., Salari, S., & Swoboda, P. (1996). *World Wide Web—Course tool: An environment for building WWW-based courses*. Fifth International World Wide Web Conference.

Half-Baked Software (1997). Markin from Creative Education Resources [Online]. Available: http://www.net-shopper.co.uk/creative/education/languages/martin/jbc.htm [1998, May 30].

McCormack, R. (1997). QuizPlease Home Page - Create quizzes and tests for the Internet - Education - Software [Online]. Available: http://www.quizplease.com/ [1998, May 30].

Shadian, R. (1998). METR&D Center Ed Tech Tools—QuizCenter [Online]. Available: http://motted.hawaii.edu/et_tools/quizcenter/ [1998, May 30].

Tinoco, L.C., Fox, E.A., Ehrich, R.W., & Fuks, H. (1996). *QUIZIT: An interactive online quiz system for WWW-based instruction*. Proceedings of the VII Brazilian Symposium on Educational Technology.