

DOCUMENT RESUME

ED 448 701

IR 020 464

AUTHOR Brusilovsky, Peter; Miller, Philip  
TITLE Web-Based Testing for Distance Education.  
PUB DATE 1999-10-00  
NOTE 7p.; In: WebNet 99 World Conference on the WWW and Internet Proceedings (Honolulu, Hawaii, October 24-30, 1999); see IR 020 454.  
PUB TYPE Information Analyses (070) -- Reports - Evaluative (142) -- Speeches/Meeting Papers (150)  
EDRS PRICE MF01/PC01 Plus Postage.  
DESCRIPTORS \*Computer Assisted Testing; \*Distance Education; Educational Technology; Evaluation Criteria; Evaluation Methods; Higher Education; Models; \*Test Construction; \*World Wide Web  
IDENTIFIERS Technology Utilization; \*Web Based Instruction

ABSTRACT

This paper provides a technology-based review of World Wide Web-based testing technologies. It suggests an evaluation framework that could be used by practitioners in Web-based education to understand and compare features available in various Web-based testing systems. In order to compare existing options, the life cycle of a question in Web-based education is divided into three stages: (1) preparation, including authoring support, storage, and selection; (2) delivery, including presentation, interaction, and getting the answer; and (3) assessment, including evaluation, grading and recording, and feedback. (Contains 33 references.) (MES)

# Web-based Testing for Distance Education

Peter Brusilovsky and Philip Miller

Carnegie Technology Education, 4615 Forbes Avenue, Pittsburgh, PA 15213, USA,  
Tel: +1-412-268-3537, Fax: +1-412-268-3505, E-mail: {plb | plm}@carnegietech.org

**Abstract:** The paper provides a technology-based review of Web-based testing technologies. It suggests an evaluation framework, which could be used by practitioners in Web-based education to understand and compare features available in various Web-based testing systems.

## 1. Introduction

Objective tests and quizzes are among the most widely used and well-developed tools in higher education. A classic test is a sequence of reasonably simple questions. Each question assumes a simple answer that could be formally checked and evaluated as correct, incorrect, or partly correct (for example, incomplete). Questions are usually classified into types by the type of expected answer. Classic types of questions includes yes/no questions, multiple-choice/single-answer (MC/SA) questions, multiple-choice/multiple-answer (MC/MA) questions, and fill-in questions with a string or numeric answer. More advanced types of questions include matching-pairs questions, ordering-questions, pointing-questions (the answer is one or several areas on a figure) and graphing-questions (the answer is a simple graph). Also, each subject area may have some specific types of questions.

Testing and quiz components were the first to be implemented and currently are the most well developed interactive components in Web-based education (WBE). Existing WBE systems differ in many aspects of dealing with tests and quizzes. When selecting a state-of-the-art technology for developing and delivering Web-based quizzes at Carnegie Technology Education we have created a multi-facet framework for comparing available systems. This paper provides a comprehensive review of features, which are important to evaluate current technologies for Web-based testing. Our framework could be used by practitioners in Web-based education to understand and compare features available in various Web-based testing systems.

## 2. Life cycle and anatomy of questions

To compare existing options we have analyzed the life cycle of a question in Web-based education (see Table 1). We divided the life cycle of a question into three stages: preparation (before active life), delivery (active life), and assessment (after active life). Each of these stages is further divided into smaller stages. For each of these stages we have investigated a set of possible support technologies.

Life of a question begins at *authoring* time. The role of WBE systems at the authoring stage is to support the author by providing a technology and a tool for question authoring. All authored questions (the content and the metadata) are *stored* in the system. The active life of a stored question starts when it is *selected for presentation* as a part of a test or quiz. This selection could be done statically by a teacher at course development time, or dynamically by a system at run time (by probability or according to some cognitive model).

Next, the system *delivers* a question: it presents the question, it provides an interface for the student to answer; it gets the answer for evaluation. At the *assessment* stage, the system should do the following things: evaluate the answer as correct, incorrect, or partly correct, deliver feedback to student, grade the question and to record student performance.

Existing WBE tools and systems differ significantly on the type and amount of support they provide on each of the stages mentioned above. Simple systems usually provide partial support for a subset of the stages. The cutting-edge systems provide comprehensive support at all the listed stages. The power of a system and the extent of provided support is seriously influenced by the level of technology used at each of the main stages - preparation, delivery and assessment. Below we will analyze the currently explored options.

| Before   | During   | After   |
|--|--|---|
| Preparation:   | Delivery:  | Assessment:   |
| <ul style="list-style-type: none"> <li>• Author</li> <li>• <u>Store</u></li> <li>• Select</li> </ul> | <ul style="list-style-type: none"> <li>• Present</li> <li>• <u>Interact</u></li> <li>• Get the answer</li> </ul> | <ul style="list-style-type: none"> <li>• Evaluate</li> <li>• <u>Grade and record</u></li> <li>• Deliver feedback</li> </ul> |

Table 1. Life cycle stages of a test question.

### 3. Preparation stage

Questions are created by a human authors - teachers and content developers. A state-of-the-art question has the following components: the question itself (or *stem*), a set of possible answers, an indication which answers are correct, a type of the interface for presentation, question-level feedback that is presented to the student regardless of the answer, and specific feedback for each of the possible answers. In addition, an author may provide metadata such as topics assessed, keywords, the part of the course a test belongs to, question weight or complexity, allowed time, number of attempts, etc. This metadata could be used to select a particular question for presentation as well as for grading the answer.

The options for *authoring support* usually depends from the technology used for storing an individual question in the system. Currently, we could distinguish two different ways to store a question: *presentation format* and *internal format*. In WBE context, storing a question in presentation format means storing it as a piece of HTML code (usually, as an HTML form). Such questions could be also called static questions. They are “black boxes” for a WBE system: It can only present static questions “as is”. The authoring of this type of questions is often not supported by a WBE system. It could be done in any of HTML authoring tools.

Storing a question in an internal format usually means storing it in a database record where different parts of the question (stem, answers, and feedback) are stored in various fields of this record. A question as seen by a student is generated from the internal format at the delivery time. Internal format opens the way for more flexibility: the same question could be presented in different forms (for example, fill-in or multiple choice) or with different interface features (for example, radio buttons or selection list). Options in multiple choice questions could be shuffled [Carbone & Schendzielorz 1997]. It provides a higher level of individualization. This is pedagogically useful and decreases the possibility of cheating. There are two major ways for authoring questions in internal format: a form-based *graphical user interface* (GUI) or a special *question markup language* [Brown 1997; Campos Pimentel, dos Santos Junior & de Mattos Fortes 1998; Hubler & Assad 1995]. Each of these approaches has its benefits and drawbacks. Currently, a GUI approach is much more popular. It is used by all advanced commercial WBE systems such as [Blackboard 1998; Question Mark 1998; WBT Systems 1999; WebCT 1999]. Note, however, that some WBE systems use GUI authoring approach but do not store questions in internal format. Instead, these systems generate HTML questions “right away” and store them as static questions

The simplest option for *question storage* is a *static test or quiz*, i.e., a static sequence of questions. The quiz itself is usually represented in plain HTML form and authored with HTML-level authoring tools. Static tests and quizzes are usually “hardwired” into some particular place in a course. One problem with this simplest technology is that all students get the same questions at the same point in the course. Another problem is that each question hardwired into a test is not reusable. A better option for question storage is a *hand-maintained pool of questions*. The pool could be developed and maintained by a group of teachers of the same subject. Each question in a question pool is usually static, but the quizzes are more flexible. Simple pool management tools let the teacher re-use questions; all quizzes may be assembled and added to the course pages when it is required. This is what we call *authoring time flexibility*. The same course next year, a different version of the course, or sometimes even different groups within the same course may get different quizzes without the need to develop these quizzes from scratch.

An even better option is to turn a hand-maintained pool into a database of questions. A database adds what we call *delivery time flexibility*. Unlike a hand-maintained pool, a database is formally structured and is accessible by the delivery system. With a database of questions not only the teacher can assemble a “quiz-on-demand”, the system itself can generate a quiz from a set of questions. Naturally, the questions could be randomly selected and placed into a quiz in a random order [Asymetrix 1998; Brown 1997; Byrnes, Debreceeny & Gilmour 1995; Carbone & Schendzielorz 1997; Ni, Zhang & Cooley 1997; Radhakrishnan & Bailey 1997; WBT Systems 1999; WebCT 1999]. As a result, all students may get personalized quizzes (a thing that a teacher can not realistically provide manually) significantly decreasing the possibility of cheating. Note that implementation of a database of questions does not require the use of a commercial database management system. Advanced university systems like QuestWriter [Bogley et al. 1996] or Carnegie Mellon Online [Rehak 1997] and many commercial systems such as TopClass [WBT Systems 1999] or LearningSpace [Lotus 1999] use full-fledge databases such as ORACLE or Lotus Notes for storing their pools of question in internal format. However, there are systems which successfully imitate a database with the UNIX file system using specially structured directories and files [Byrnes, Debreceeny & Gilmour 1995; Gorp & Boysen 1996; Merat & Chung 1997].

A problem for all systems with computer-generated quizzes is how to ensure that these quizzes include a proper set of questions. The simplest way to achieve it is to organize a dedicated question database for each lesson. This approach, which is, for example, used in WebAssessor [ComputerPREP 1998], reduce question reusability between

lessons. More advanced systems like TopClass [WBT Systems 1999] can maintain multiple pools of and can use several pools for generating each quiz. With this level of support a teacher can organize a pool for each topic or each level of question complexity and specify the desired number of questions in a generated quiz to be taken from each pool.

A database of questions in internal format is currently a state-of-the-art storage technology. Research teams are trying to advance it in three main directions. One direction is related to *parameterized questions* as in CAPA [Kashy et al. 1997], EEAP282 [Merat & Chung 1997], or Mallard [Brown 1997; Graham, Swafford & Brown 1997]. This allows one to create an unlimited number of tests from the same set of questions and can practically eliminate cheating [Kashy et al. 1997]. The second direction of research is related to question metadata. If the system knows a little bit more *about* the question (for example, type, topics assessed, keywords, part of the course a test belongs to, weight or complexity) then the system can generate customized and individualized quizzes by author's or system's request. This means that the authors could specify various parameters for the quiz their student needs at some point of the course: total number of questions, proportion of questions of specific types or for specific topics, difficulty, etc., and the system will generate a customized quiz on demand (that is still randomized within the requirements) [Byrnes, Debreceeny & Gilmour 1995; Merat & Chung 1997; Rehak 1997; Rios, Pérez de la Cruz & Conejo 1998]. This option is definitely more powerful than simple randomized quizzes. Systems that make extensive use of metadata really "know" about the questions and their functionality. The third direction of research is the adaptive sequencing of questions. This functionality is based on an overlay student model which separately represents student knowledge of different concepts and the topics of the course. Intelligent systems such as ELM-ART [Weber & Specht 1997], Medtec [Eliot, Neiman & Lamar 1997], [Lee & Wang 1997], SIETTE [Rios, Pérez de la Cruz & Conejo 1998], Self-Learning Guide [Desmarais 1998] can generate challenging questions and tests adapted to the student level of knowledge as well as reduce the number of questions required to assess the students state of knowledge.

#### 4. Delivery stage

The interaction technology used to get an answer from the student is one of the most important parameters of a WBE system. It determines all delivery options and influences authoring and evaluation. Currently, we distinguish five technologies: HTML links, HTML/CGI forms, scripting language, plug-in, and Java.

HTML links is a simple interaction technology that presents a set of possible answers as list of HTML links. Each link is connected to a particular feedback page. The problems here are that questions are hard to author (because question logic must be hardwired into course hypertext) and that it supports only yes/no and MC/SA questions. This technology was in use in the early days of WBE when more advanced interaction technologies like Common Gateway Interface (CGI), JavaScript or Java were not established [Holtz 1995].

The most well-established technology for Web testing which is used now in numerous commercial and university-grown systems is a combination of HTML forms and CGI-compliant evaluation scripts. HTML forms are very well suited for presenting main types of questions. Yes/no and MC/SA questions are represented by radio buttons, selection lists, pop-up menus, MC/MA questions are represented by multiple selection lists or checkboxes. Fill-in questions are implemented with input fields. More advanced questions such as matching pairs or ordering can also be implemented using forms. In addition, hidden fields can be successfully used to hold additional information about the test which a CGI script may need. There are multiple benefits of using server-side technology such as form/CGI technology and a similar server-side map technology that can be used for implementing graphical pointing questions. Test development is relatively simple and can even be done with HTML authoring tools. Sensitive information which is required for test evaluation (such as question parameters, answers, feedback) may be safely stored on the client side preventing students from stealing the question (the only external information which is required in a well-developed system to evaluate a test is the test ID and the student ID). Server-side evaluation makes all assessment time functions (such as recording results, grading, providing feedback) easy to implement. All these functions could be performed by the same server-side evaluation script. The main problem of server-side technology is its low expressive power. It is well suited only for presenting basic types of tests. More advanced types of tests as well as more interactive types of tests (for example, tests which involve drag-and-drop activities) can not be implemented with pure server-side technology. Authoring questions with server-side evaluation is tricky because a question's functionality is spread between its HTML presentation (either manually authored or generated) and a CGI evaluation script. Another serious problem is CGI-based questions do not work when a user's connection to the server is broken or very slow.

A newer technology for question delivery and evaluation is JavaScript [McKeever, McKeever & Elder 1997]. The interface provided by the JavaScript interaction technology is similar to the one of form/CGI technology. At the same time, JavaScript functionality supports more advanced, interactive questions, for example, selection of a

relevant fragment in a text. With pure JavaScript technology all data for question evaluation and feedback as well as evaluation program should be stored as a part of the question text. It means that a JavaScript question can work in standalone mode. It means that the question is self-sufficient: everything for presentation and evaluation is in the same file, and is a very attractive option for authoring. But it also means that students can access the source of the question and crack it. Also, with pure JavaScript evaluation technology there is no way for recording the results and grades. With all the above features JavaScript technology is a better choice for self-assessment tests than for assessments used in grading. We think the proper place for JavaScript in WBE is in a hybrid JavaScript/server technology. With this technology JavaScript can be used to present more types of questions, do it more interactively and with compelling user interfaces leaving evaluation and recording to be done by traditional CGI for reasons of security [ComputerPREP 1998; WebCT 1999].

A higher level of interface freedom can be achieved by using a plug-in technology.<sup>1</sup> The only example of serious use of this technology in education is the Shockwave plug-in [Macromedia 1998] which can run multimedia presentations prepared with several Macromedia authoring tools. Currently, Shockwave technology is used in WBE mainly for delivering “watch-only” animations, but this technology is more powerful. In fact, a variety of very attractive Shockwave-deliverable questions could be developed using Macromedia tools with relatively low effort. Some examples could be provided by Medtec [Eliot, Neiman & Lamar 1997]. The negative side is the same as with JavaScript: recording assessment results requires connection to the server. Until recently, Shockwave provided no Internet functionality and its users had to apply special techniques (e.g. saving evaluation results in a temporal file). Due to Shockwave communication problems, some teams that started with Shockwave migrated later to more powerful Java technology [Eliot, Neiman & Lamar 1997]. Still Shockwave still stands as a solid (and overlooked) platform for delivering various self-assessment questions.

The highest level technology for question delivery is provided by Java. An important advantage of Java is that it is a complete programming language designed to be integrated with browser functionality and the Internet. Java combines connectivity of form/CGI technology and the interactivity of Shockwave and JavaScript. Any question interface can be developed with Java, and, at the same time, Java-made questions can naturally communicate with the browser as well as with any Internet object (a server or a Java application). Examples of systems which heavily use Java-based questions are FLAX [Routen, Graves & Ryan 1997], NetTest [Ni, Zhang & Cooley 1997], Mallard [Graham & Trick 1997], and Medtec [Eliot, Neiman & Lamar 1997]. Developing question interfaces with Java is more complicated than with form/CGI technology and it is not surprising that all the examples mentioned above were produced by advanced teams of computer science professionals. However, the complexity will not stop this technology. Java is currently the way to implement a variety of question types non-implementable with form/CGI technology such as multiple pointing questions, graphing questions, and specialized types of questions. Developing Java-based questions can become suitable for ordinary authors with the appearance of Java based authoring systems [Ni, Zhang & Cooley 1997; Routen, Graves & Ryan 1997].

## 5. Assessment stage

As we noted, the choice of interaction technology significantly influences *evaluation* options. Evaluation is the time when an answer is judged as correct, incorrect, or partially correct (for example, incomplete). Usually, correct and incorrect answers are provided at authoring time, so evaluation is either hardwired into the question like in MC/SA questions, or performed by simple comparison (in fill-in questions). There are very few cases that require more advanced evaluation technology. In some domains correct answers may not be literally equal to a stored correct answer. Examples are a set of unordered words, a real number, a simple algebraic expression [Holtz 1995; Hubler & Assad 1995]. In this situation a simple comparison program is required. Some systems may apply special intelligent technologies for matching answers [Hubler & Assad 1995]. Finally, in some cases a “domain expert” such as the Lisp interpreter for Lisp programming as in the ELM-ART system [Brusilovsky, Schwarz & Weber 1996] or a computer algebra system for algebra domain [Pohjolainen, Multisilta & Antchev 1997] is required to evaluate the answer. The first two evaluation options are very simple and could be implemented with any interface technology - even JavaScript could be used to write a simple comparison program. If more advanced computation is required (as in the case of intelligent answer matching) the choice is limited to full-function programming with either Java or a server side program using a CGI interface. If a “domain expert” is required for evaluation, the only option currently is to run a domain expert on the server side with a CGI-compliant gateway. In fact, a number of “domain expert” systems (for example, Mathematica computer algebra system) have a CGI gateway.

---

<sup>1</sup> Plug-in technology enables independent vendors to extend the browser functionality by developing specially structured programs called *plug-ins*. At start-up time, a browser loads all plug-ins located in a special directory and they become parts of the browser code.

The usual options for the *feedback* include: simply telling if the answer is correct, not, or partially correct, giving correct answer, and providing some individual feedback. Individual feedback may communicate: what is right in the correct answer, what is bad in incorrect and partially incorrect answer, provide some motivational feedback, and provide information or links for remediation. All individual feedback is usually authored and stored with the question. A system that includes assessed concepts or topics as a part of question metadata can provide good remedial feedback without direct authoring since it “knows” what knowledge is missing and where it can be found. It means that the power of feedback is determined by authoring and storage technology. The amount of information presented as feedback is determined by the context. In self-assessment the student usually receives all possible feedback - the more the better. This feedback is a very important source of learning. In a strict assessment situation the student usually gets neither a correct answer, nor whether the answer is correct. The only feedback for the whole test might be the number of correctly answered questions in a test [Rehak 1997]. This greatly reduces the student’s chances for cheating and student’s chances to learn. To support learning, many existing WBE systems make assessment less strict and provide more feedback trying to fight cheating by other means. The only way to combine learning and strict assessment is to use more advanced technologies such as parameterized questions [Brown 1997; Hubler & Assad 1995; Kashy et al. 1997; Merat & Chung 1997] and knowledge based test generation [Eliot, Neiman & Lamar 1997; Weber & Specht 1997] which can generate an unlimited number of questions. In this situation a WBE system can provide full feedback without promoting cheating.

If a test is performed purely for self-assessment then generating feedback could be the last duty of a WBE system in the “after-testing” stage. The student is the only one who needs so see test results. In the assessment context the last duty of a WBE system in the process of testing is to *grade student performance on a test and to record these data* for future use. Grades and other test results are important for teachers, course administrators, and students themselves (a number of authors noted that the ability to see their grades online is the most student-appreciated feature of a WBE system). Early WBE systems provided very limited support for a teacher in test evaluation. Results were either sent to the teacher by e-mail or logged into a special file. In both cases a teacher was expected to complete grading and recording personally: to process test results and grade them, to record the grades, and to ensure that all involved parties get access to data according university policy. This option is easy to implement and it does not require that teachers learn any new technology. For the latter reason this technology is still used as an option in some more advanced systems [Carbone & Schendzielorz 1997]. However, a system that provides no other options for grading and recording is now below a state-of-the-art. A state-of-the-art WBE system should be able to grade a test automatically, recording test results in a database. It also should provide properly restricted access to the grades for students, teachers, and administrators. Restrictions are usually determined by university policies. For example, a student may not be allowed to see grades of other students or a teacher could be allowed to change the automatically assigned grades. Many university-level systems [Bogley et al. 1996; Brown 1997; Carbone & Schendzielorz 1997; Gorp & Boysen 1996; Hubler & Assad 1995; MacDougall 1997; Ni, Zhang & Cooley 1997; Rehak 1997] and almost all commercial level systems [Lotus 1999; WBT Systems 1999; WebCT 1999] provide this option in a more or less advanced way. Less advanced systems usually store the grades in structured files and provide limited viewing options. Advanced systems use database technology to store the grades and provide multiple options for viewing the grades and other test performance results such as time on a test or a number of efforts made. Database technology makes it easy to generate various test statistics involving results of many students on many course tests. In a Web classroom, where student-to-student and student-to-teacher communication is limited, comparing statistics is very important for both - teachers and students to get the “feeling” of the classroom. For example, by comparing class average with personal grades a student can determine class rank. By comparing class grades for different tests and questions a teacher can find too simple, too difficult, and even incorrectly authored questions.

## References

- [Asymetrix 1998] Asymetrix (1998). Librarian, Asymetrix Learning Systems, Inc., Bellevue, WA. <http://www.ASYMETRIX.com/products/librarian/> (Accessed 20 September, 1998)
- [Blackboard 1998] Blackboard (1998). CourseInfo 1.5, Blackboard Inc. [http://www.blackboard.net/ps\\_courseinfo.htm](http://www.blackboard.net/ps_courseinfo.htm) (Accessed 21 August, 1998)
- [Bogley et al. 1996] Bogley, W.A. et al. (1996). New pedagogies and tools for Web based calculus. WebNet'96, World Conference of the Web Society, AACE. 33-39.
- [Brown 1997] Brown, D.J. (1997). Writing Web-based questions with Mallard. FIE'97, Frontiers in Education Conference, Stipes Publishing L.L.C. 1502.
- [Brusilovsky, Schwarz & Weber 1996] Brusilovsky, P., Schwarz, E., & Weber, G. (1996). ELM-ART: An intelligent tutoring system on World Wide Web. In Frasson, C., Gauthier, G., & Lesgold, A. (Ed.), Intelligent Tutoring Systems (Lecture Notes in Computer Science, Vol. 1086). Berlin: Springer Verlag. 261-269.

- [Byrnes, Debreceeny & Gilmour 1995] Byrnes, R., Debreceeny, R., & Gilmour, P. (1995). The Development of a Multiple-Choice and True-False Testing Environment on the Web. Ausweb95: The First Australian World-Wide Web Conference, Southern Cross University Press. <http://elmo.scu.edu.au/sponsored/ausweb/ausweb95/papers/education3/byrnes/>
- [Campos Pimentel, dos Santos Junior & de Mattos Fortes 1998] Campos Pimentel, M.d.G., dos Santos Junior, J.B., & de Mattos Fortes, R.P. (1998). Tools for authoring and presenting structured teaching material in the WWW. WebNet'98, World Conference of the WWW, Internet, and Intranet, AACE. 194-199.
- [Carbone & Schendzielorz 1997] Carbone, A., & Schendzielorz, P. (1997). Developing and integrating a Web-based quiz generator into the curriculum. WebNet'97, World Conference of the WWW, Internet and Intranet, AACE. 90-95.
- [ComputerPREP 1998] ComputerPREP (1998). WebAssessor, ComputerPREP, Inc, Phoenix, AZ. <http://www.webassessor.com> (Accessed 23 May, 1998)
- [Desmarais 1998] Desmarais, M.C. (1998). Self-Learning Guide Stuttgart, Germany, CRIM, Montreal. <http://www.crim.ca/hci/demof/gaa/introduction.html> (Accessed July 5, 1999)
- [Eliot, Neiman & Lamar 1997] Eliot, C., Neiman, D., & Lamar, M. (1997). Medtec: A Web-based intelligent tutor for basic anatomy. WebNet'97, World Conference of the WWW, Internet and Intranet, AACE. 161-165.
- [Gorp & Boysen 1996] Gorp, M.J.V., & Boysen, P. (1996). ClassNet: Managing the virtual classroom. WebNet'96, World Conference of the Web Society, AACE. 457-461.
- [Graham, Swafford & Brown 1997] Graham, C.R., Swafford, M.L., & Brown, D.J. (1997). Mallard: A Java Enhanced Learning Environment. WebNet'97, World Conference of the WWW, Internet and Intranet, AACE. 634-636.
- [Graham & Trick 1997] Graham, C.R., & Trick, T.N. (1997). An innovative approach to asynchronous learning using Mallard: Application of Java applets in a freshman course. FIE'97, Frontiers in Education Conference, Stipes Publishing L.L.C. 238-244.
- [Holtz 1995] Holtz, N.M. (1995). The Tutorial Gateway, Carleton University, Ottawa, CA. <http://www.civeng.carleton.ca/~nholtz/tut/doc/doc.html> (Accessed 1995)
- [Hubler & Assad 1995] Hubler, A.W., & Assad, A.M. (1995). CyberProf: An Intelligent Human-Computer Interface for Asynchronous Wide Area Training and Breakching. 4th International World Wide Web Conference. <http://www.w3.org/pub/Conferences/WWW4/Papers/247/>
- [Kashy et al. 1997] Kashy, E. et al. (1997). Using networked tools to enhance student success rates in large classes. FIE'97, Frontiers in Education Conference, Stipes Publishing L.L.C. 233-237.
- [Lee & Wang 1997] Lee, S.H., & Wang, C.J. (1997). Intelligent hypermedia learning system on the distributed environment. ED-MEDIA/ED-TELECOM'97 - World Conference on Educational Multimedia/Hypermedia and World Conference on Educational Telecommunications, AACE. 625-630.
- [Lotus 1999] Lotus (1999). LearningSpace 2.0, Lotus, Cambridge, MA. <http://www.lotus.com/products/learningspace.nsf> (Accessed 5 June, 1999)
- [MacDougall 1997] MacDougall, G. (1997). The Acadia advantage adademic development centre and the authomatic courseware management systems. ED-MEDIA/ED-TELECOM'97 - World Conference on Educational Multimedia/Hypermedia and World Conference on Educational Telecommunications, AACE. 647-652.
- [Macromedia 1998] Macromedia (1998). Shockwave, AACE. <http://www.macromedia.com/shockwave/> (Accessed September 12, 1998)
- [McKeever, McKeever & Elder 1997] McKeever, S., McKeever, D., & Elder, J. (1997). An authoring tool for constructing interactive exercises. WebNet'97, World Conference of the WWW, Internet and Intranet, AACE. 695-696.
- [Merat & Chung 1997] Merat, F.L., & Chung, D. (1997). World Wide Web approach to teaching microprocessors. FIE'97, Frontiers in Education Conference, Stipes Publishing L.L.C. 838-841.
- [Ni, Zhang & Cooley 1997] Ni, Y., Zhang, J., & Cooley, D.H. (1997). NetTest: An integrated Web-based test tool. WebNet'97, World Conference of the WWW, Internet and Intranet, AACE. 710-711.
- [Pohjolainen, Multisilta & Antchev 1997] Pohjolainen, S., Multisilta, J., & Antchev, K. (1997). Matrix algebra with hypermedia. Education and Information Technologies, 1 (2), 123-141.
- [Question Mark 1998] Question Mark (1998). Perception, Question Mark Corporation, Stamford, CT. <http://www.questionmark.com/> (Accessed September 30, 1998)
- [Radhakrishnan & Bailey 1997] Radhakrishnan, S., & Bailey, J.E. (1997). Web-based educational media: Issues and empirical test of learning. WebNet'97, World Conference of the WWW, Internet and Intranet, AACE. 400-405.
- [Rehak 1997] Rehak, D. (1997). A database architecture for Web-based distance education. WebNet'97, World Conference of the WWW, Internet and Intranet, AACE. 418-425.
- [Rios, Pérez de la Cruz & Conejo 1998] Rios, A., Pérez de la Cruz, J.L., & Conejo, R. (1998). SIETTE: Intelligent evaluation system using tests for TeleEducation. Workshop "WWW-Based Tutoring" at 4th International Conference on Intelligent Tutoring Systems (ITS'98). <http://www-aml.cs.umass.edu/~stern/webits/itsworkshop/rios.html>
- [Routen, Graves & Ryan 1997] Routen, T.W., Graves, A., & Ryan, S.A. (1997). Flax: Provision of interactive courseware on the Web. Cognition and the Web '97. 149-157. <http://www.cms.dmu.ac.uk/coursebook/flax/>
- [WBT Systems 1999] WBT Systems (1999). TopClass 3.0, WBT Systems, Dublin, Ireland. <http://www.wbtsystems.com/> (Accessed 5 July, 1999)
- [WebCT 1999] WebCT (1999). World Wide Web Course Tools 1.3.1, WebCT Educational Technologies, Vancouver, Canada. <http://www.webct.com> (Accessed 15 February, 1999)
- [Weber & Specht 1997] Weber, G., & Specht, M. (1997). User modeling and adaptive navigation support in WWW-based tutoring systems. In Jameson, A., Paris, C., & Tasso, C. (Ed.), User Modeling. Wien: Springer-Verlag. 289-300.



**U.S. Department of Education**  
Office of Educational Research and Improvement (OERI)  
National Library of Education (NLE)  
Educational Resources Information Center (ERIC)



## NOTICE

### REPRODUCTION BASIS



This document is covered by a signed "Reproduction Release (Blanket) form (on file within the ERIC system), encompassing all or classes of documents from its source organization and, therefore, does not require a "Specific Document" Release form.



This document is Federally-funded, or carries its own permission to reproduce, or is otherwise in the public domain and, therefore, may be reproduced by ERIC without a signed Reproduction Release form (either "Specific Document" or "Blanket").