Developing Adaptive Intelligent Tutoring Systems: A General Framework and Its Implementations

Jens O. Liegle[1]
and
Han-Gyun Woo

Department of Computer Information Systems, Georgia State University
Atlanta GA 30303, USA

Abstract

Web-based training is increasingly gaining popularity both in industry and education. Although a number of studies, experiments, and developments have been conducted in this area, few evidence cases of success have been reported. One likely reason for the lack of success is that just placing lecture notes on the web does not train. This situation can be improved through the use of training software such as Intelligent Tutoring Systems (ITS). ITS incorporate built-in expert systems in order to monitor the performance of a learner and to personalize instruction on the basis of adaptation to learners' learning style, current knowledge level, and appropriate teaching strategies.

However, researchers and developers quickly find out that developing such systems is an enormous task, which is further complicated by the fact that one cannot simply borrow tools from other systems and incorporate them due to various levels of incompatibility at the programming and knowledge base level. To allow for more general ITS, which means that it can be used in other domains, it is required that ITS should be designed and implemented so as to support easy modification of lecture content, modification of decision rules in the expert system, and to support various methods to measure the performances of learning.

In this paper, we propose a general framework and data model for web-based adaptive ITS that allows knowledge to be stored in such a way that is not only independent of the knowledge domain, but also supports the storage of transfer knowledge relationships and prerequisite knowledge relationships. We expect that our unified approach could contribute to the establishment of cumulative research traditions in ITS studies.

Keywords: Intelligent Tutoring Systems, Cognitive Style, Learning Style, System Development, Framework

1. INTRODUCTION

Computer-based training (CBT) has become more popular since the 1990s. Corporations delivered 10% of their training through CBT Systems in 1996, and 16% of the Fortune 1000 companies trained through multimedia systems (Filipczak, 1996). These days, web-based training is increasingly gaining popularity in education as well as in industry. A recent survey found that web-based training (WBT) is the newest and fastest growing training method (McGee, 1998), a trend that is predicted to result in an online training industry reaching $28 billion per year worldwide by 2001 (Dillon, 1997). A reason for this trend is that the Web is a very cost-effective way of distributing training material to employees (Liegle and Madey, 1997; McGee, 1998; Dede, 1996). Universities are following this web-training trend by offering entire classes and degrees online, e.g., New York University, Drexel University, Britain's Open University, and the University of Phoenix (Herther, 1997).

One of the benefits from web-based training is punctuality. Using web-based training, corporations are able to deliver just-in-time training when it is needed by employees (Gibbons, 1997). Moving to the Web provides benefits in form of on-the-job continuing education and general training (Moore, 1996). Furthermore, web-based training is platform independent and makes maintenance of systems and contents much easier (Cole, et al., 1997).

[1] jliegle@cis.gsu.edu

Despite the fact that many studies, experiments, and prototype developments have been conducted in the area of CBT systems, and although many systems have been developed that support training and lecturing in the form of CAL, CAI, CBT, and WBT (Hawkins, 1997; Kruse, 1997; Hardaway and Will, 1997), the vast majority of such systems in industry and commerce are not as successful as had been hoped (Martinsons and Schindler, 1995; Nelson, Whitener, and Philcox, 1995; Wells, Layne and Flowers, 1993; and Major and Reichgelt, 1992).

One important reason for this lack of success is that "information is not instruction" (Jim L'Allier, in Bernstein, 1998 p.16) and "information is not training" (Schank 1994, p. 634), which means that even hypermedia presentations do not teach by themselves, but instead only present information. Schank, Korcuska, and Jona (1995) stated that pure information presentation fails as a technique for education, and they criticized page turning, 'next button' oriented multimedia applications for not supporting learning.

A potential solution to this problem is the use of training software such as Intelligent Tutoring Systems (ITS) with built-in artificial intelligence. These systems, which adapt themselves to the current knowledge stage of the learner and support different learning strategies on an individual basis, could be integrated with the Web (Nkambou and Gauthier, 1996). However, the nature of ITS research requires various technologies and different disciplinary knowledge, making the reuse of previous studies relatively difficult compared to other research areas. To solve this problem, we adopt a unified approach for adaptation of learners' current knowledge and learning styles and effective teaching strategies, combined with an object-oriented approach for system development and maintenance. In the following section, we first discuss benefits of adaptive ITS with extensive review of previous research. Then, we suggest a general framework and data model for web-based adaptive ITS which can be used in other domains. Finally, we present our implementation for Pl@tos, an adaptive ITS that has been developed in order to teach introductory courses of computer programming languages, and show how to combine a reasoning expert system shell with lecture content in ITS. In doing this, we expect that our unified and object-oriented approach could contribute to the establishment of cumulative research traditions in ITS studies.

2. BENEFITS OF ADAPTIVE INTELLIGENT TUTORING SYSTEMS

Web-based ITS are still in the early developmental and experimental stages and much research will need to be done at all levels to evaluate the quality of learning (Herther, 1997). Thus, there is a need for exploration and evaluation, which makes web-based education a hot research and development area (Khan, 1997).

Web-based educational systems have the benefits of classroom and platform independence. Adaptation is especially important for such systems because web-based applications are used by a much wider variety of users than standalone applications, and because in many cases the user is alone at home or in the office, without support from human instructors or peers (Brusilovsky, 1998). As an additional benefit, adapting the instructional methods and the training material to individual learning styles has led to improved learner performance (Bostrom, et al., 1990).

Figure 1 shows an overview of the instruction process. A teaching strategy is selected based on the knowledge type being taught and the learner's current depth of knowledge. The actual instruction then changes the learner's level of knowledge depth by either adding new knowledge or "deepening" existing knowledge. Testing is conducted to evaluate the success of the instruction and the potential need for a repetition or modification of instruction.

Traditional classrooms and conventional CBT systems face a number of problems: In Figure 1, (*) indicates that there can be multiple learners at the same time, e.g. in a classroom. Each of these learners will have a different depth of knowledge, and thus will require different types of instruction. These differences will change over time, since the change in depth of knowledge through instruction (**) will vary from person to person. In addition, learners will have their respective preferred learning styles, and a typical lecture is limited to either repeating the same material over and over again in different formats trying to support different learning styles, or ignores these differences altogether.
In order to show learners whether they understood the material, tests are given to them. However, time

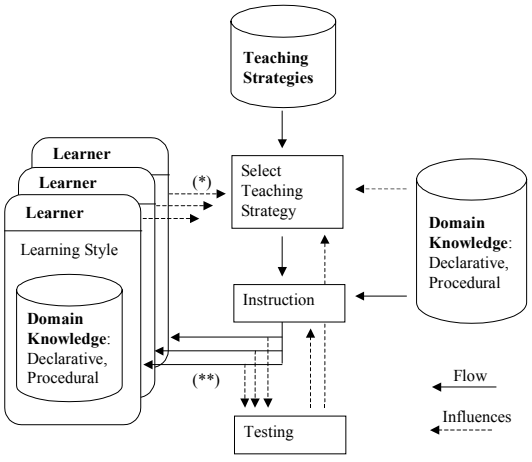

Figure 1: Need for Personalized Instruction

limitations usually prevent the instructor from repeating the instruction to those individuals who failed parts of the test. Instead, grades are used to inform the student (and potential employers) that a particular knowledge type has not been mastered. At this point, the student alone is responsible for changing the situation and obtaining mastery, without the help of an instructor.

A better instruction method would be to personalize the instruction based on the background and the progress of each individual student via either human or computer-based tutors. A personalized method of instruction is characterized by 1) learner-controlled pacing, 2) the ability to retake tests until mastery is demonstrated, 3) immediate feedback, 4) small units of instructional material, 5) the use of peer proctors to administer feedback and testing, and 6) optional lectures (Calhoun, 1975). For computer-based training systems, the computer can take over the role of peer proctors in form of online-testing and grading, and lectures can be placed online as well. The other principles can be directly applied to the development of ITS by providing learners with different versions, different difficulty levels, etc., based on their performance and previous knowledge.

In the following sections, we review the two approaches that have emerged in the area of software training research (Olfman and Mandviwalla, 1994) that try to improve end-user training: 1) adaptation of the training material content and 2) adaptation of the training material presentation.

Adaptation based on Content

Adaptation of the content implies that different learners receive different versions of the same lecture. Common approaches are 1) adapting the level of difficulty appropriate to level of understanding, 2) adapting the level of depth according to stated learning goal of the learner, which can range from high level overview to in-depth discussions, 3) adding appropriate references to existing knowledge, e.g. references to "loops" in Visual Basic when learning "loops" in C++ for only those learners that know Visual Basic (Merrill, et al., 1996).

Content adaptation has also the benefit of allowing the ITS to support a variety of learning styles such as textual vs. visual or audio presentation forms of the same training material, or even different versions of the content to support learning styles such as concrete vs. abstract oriented learners or analytics vs. wholists by providing different content to different learners (Davidson, et al., 1992; Egan, 1988; Gitomer, 1987; Knarr, 1996; Kolb, 1976; Sadler-Smith, 1997).

Adaptation Based on Sequence

Besides adapting the content of the training material, another well-researched area is the adaptation of the instruction sequence. Bernstein (1998) found that two approaches to lesson sequencing seem to be work well on the Web. These approaches are based on either behavioralism or constructionism theory (Brandt, 1997). The former guides a student through predefined steps (system control), and the latter provides all the resources and lets students construct knowledge themselves (learner control).

Niemiec, Sikorski, and Walberg (1996) conducted a meta-analysis of the research on learner-control. They found that most studies used a combination of learner-control features, had a system-controlled control group, and administered an immediate posttest. At first glance, the overall average result of near zero effect of learner- vs. system-control might surprise researchers and discourage any further investigation in this area. However, the way research on learner-control has been conducted has been criticized (Reeves, 1993), and any meta-analysis in this area in fact compares "apples and oranges" and thus has to be examined very closely. The following paragraphs present an overview of research in this area.

Schnackenberg et al. (1998) presented a thorough overview of the research on learner- vs. system-control. Arguments in favor of learner-control are that 1) learners know their own instructional needs best, 2) learner-control can help students become independent learners, and 2) learners construct their own knowledge in the context of their own needs and experiences and require control over the learning process to do so. Critics claim that that learner-control distracts learners because it forces them to interrupt their learning and pay attention to the sequencing of material (Schnackenberg, et al., 1998). Others (Murray, 1998) claim that beginners are unable to make the right sequence choices, because they lack the background knowledge necessary to make educated decisions. Similarly, Lieberman and Linn (1991) found that novices should benefit from system-control, but more advanced students could benefit from learner-control. Others go even further claiming that the degree of learner-control should depend on the learner's familiarity with the topic as well as the learner's motivation, aptitude, and attitude (Merrill, et al., 1992).

In order to avoid the constant questions about how to proceed in learner-control systems, an experiment by Schnackenberg et al. (1998) tried asking the learners in advance about the amount of instruction and practice they desired. Matching and mismatching the learners to their preferred amounts, the study showed that although subjects preferred the lean version of the instructional material, scores were higher for subjects who took the full version, which shows that learners are not making the best choices when asked directly.

Most learner-control studies ignore the proficiency level of the learners. For example, McGrath (McGrath, 1992) examined the question of whether learners do benefit from learner-controlled systems. She examined the

impact of hypertext, CAI, paper, and program-control and found differences in score, navigation, and time spent. These results must be examined carefully since Schröder, Möbus, and Pitschke (1995) found that novice learners used a fairly passive strategy for moving through a hypermedia system, not utilizing their selection control and instead following a linear viewing pattern. On the other hand, Rieman, Young, and Howes (1996) showed in their experiments that students did not follow a linear viewing pattern when they had full control; instead, no specific sequence was followed in what they call exploratory learning. Supporting both these views, Allinson (1992) reported in her study that some subjects used a more linear navigation approach, and others preferred self-determined hypertext navigation. The effect of navigation on performance was examined by Melara (1996) who found no performance differences when students used a hierarchical organized system compared to a network structure. However, others like Tennyson (1981) and later Goforth (1994) found performance differences in that learner control is more effective than system-control, but Young (1996) supported this finding only for learners with high self-regulated learning strategies, not for others, showing a potential link of navigation habits and level of expertise. Based on this discussion, the research on the effectiveness of learner- vs. system-control seems to be inconclusive at best.

It might be the case that the two approaches, learner-control and system-control, work well for different students, but at this point, the research regarding the optimal balance of learner- vs. system-control is inconclusive. It would appear that the best system would adapt itself to the learner by providing both the behavioralist and the constructivist approach based on the learner's preference and personality type (Tan, 1996).

One such experiment was conducted by Melara (1996), who examined the effect of learning style (based on Kolb's Learning Style Inventory) on learner performance within two different hypertext structures: hierarchical and network. Her experiment showed no significant differences in achievement for Explorers and Observers using either hypertext structure. She raises the following point: contrary to Observers, Explorers are expected to prefer experimentation to observation. Studies are needed that examine the time spent on different activities that are targeted towards these two different personality types (Melara, 1996). Melara's study comparing two versions of learner-control only examined the effect of different content structure and navigation. A more interesting study would be to examine the effect of the Explorer vs. Observer personality-type dimension on system- vs. learner-controlled sequence.

3. THE PL@TOS INTELLIGENT TUTORING SYSTEM

Our research has concentrated on building an adaptive intelligent tutoring system that teaches the principles of structured programming (Liegle, 1999). In this paper, we will present the steps we are taking in order to make the system adaptable to other domains as well. One of our objectives is to provide a general framework for ITS (see Figure 2) and its prototype (the Programming L@nguage TutOring System - Pl@tos). To be a more general ITS, which means that it can be used in other domains, the ITS needs to be designed and implemented so as to support modification of a) the lecture content, b) the decision rules and the fact base of the expert system, and c) the methods to measure performances of learning. The most common stumbling block in ITS research is that the reuse of previous studies is relatively difficult compared to other research areas due to the nature of ITS that involve various technologies and the differences of disciplinary knowledge. Therefore, we expect that our unified and object-oriented approach about system implementation could contribute to the establishment of cumulative research traditions in ITS studies.

The goal of Pl@tos is to give learners personalized instruction based on their preferred learning style, background knowledge, and current understanding of the material. The following paragraphs review how the expert system component of Pl@tos selects the most appropriate path of action, the design of the knowledge repository, and the implementation environment of the new version of Pl@tos.

Figure 2 shows how information about a user and the knowledge repository are used by the expert system to identify the most appropriate content with the best teaching strategy for a given user. The expert system identifies what should be learned next by comparing the contents of a given course (see Figure 3) to the current knowledge of the learner (see History, Figure 3). Using the prerequisite relationship between different topics,

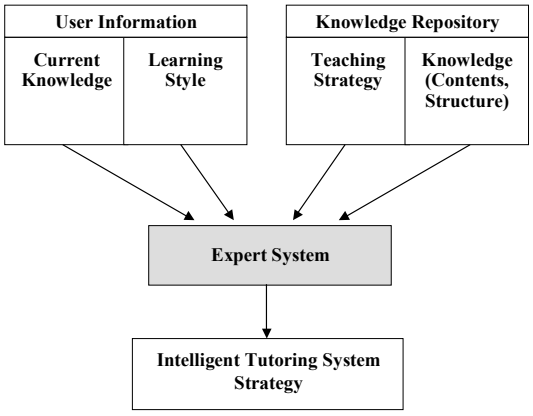

Figure 2: Integration of an Expert System Shell

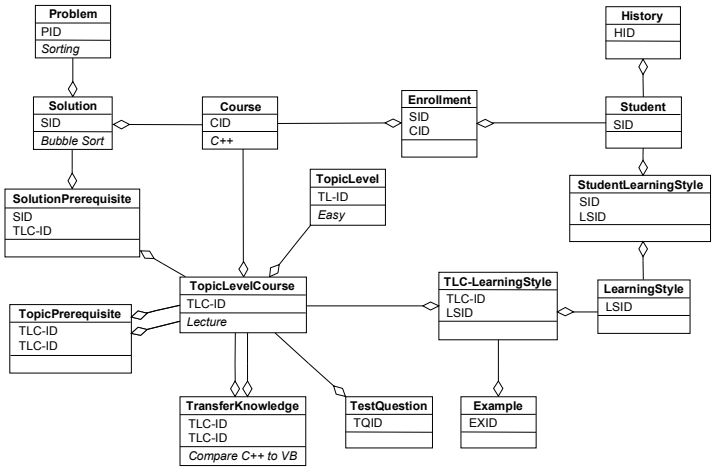

Figure 3: Top-Level Data Model

the system identifies a set of recommended lectures and depending on the learning style either presents one of them or asks the learner to select one. The learning style is used to determine the most appropriate teaching strategy for a given lecture and student by using the TLC-Learning style relationship (see Figure 3).

Figure 3 presents a top-level data model of the system. A student can enroll in multiple courses, and each student can have multiple preferred learning styles. To simplify the design of the system, the history table acts like a log-file for each student and keeps track of which test, exercise, etc. students took and whether they passed or failed. Each course has a series of topics or lectures. Each topic is available at different difficulty levels, and the most appropriate level is presented to the students based on past performance. Each topic can have multiple special versions targeted for the various learning styles, e.g. a graphical representation for visually oriented learners. There can also be different versions of the topic based on the depth of learning, from high-level introduction and overviews to in-depth discussions. Associated to each topic are test questions and a list of required prerequisite knowledge. This later relationship is used to determine what topic the student can/should learn next by only presenting topics where all prerequisite topics have been already mastered. The transfer-knowledge table links topics from different courses together. An example would be if a student already knows how to program a loop in Visual Basic and is learning about loops in C++, then the additional information about similarities and differences between the two languages can be presented. This special information would only confuse students that do not know Visual Basic, thus this information is stored separately.

A unique approach was used to store and identify exercises/problems. Each problem can have multiple solutions, and each solution has a list of required prerequisite knowledge associated. When students that learn for example loops in C++ and want to practice their skill on a problem, the system can search through all solution-prerequisites and identify those where a particular student has already mastered all but the currently practiced topic and present them to the student.

The implementation of different teaching strategies is challenging. While some researchers have suggested that general teaching strategies indeed exist (Merrill, 1991; Merrill, et al., 1996; Merrill and Li, 1989), our own research has shown that the implementation of such strategies for particular domains such as computer programming are not easily transferable to other domains (Liegle, 1999)[2]. We are currently redesigning the implementations for teaching strategies such as the "show", "demonstrate", "is-part-of", and others for more general use.

Figure 4 presents the current physical environment of the Pl@tos system. In implementing the Pl@tos system, we adopted an object-oriented approach in order to ease the reuse of components, content, or even whole systems. The main parts of systems with the exception of the web server and the database are developed with Java, a pure object-oriented programming language. The Pl@tos prototype is implemented on a LINUX web server running apache. Instead of CGI programs, it uses Java Servlets to connect the system to the database and JESS. JESS is an expert system shell providing a rule engine and scripting environment. JESS is implemented as a set of Java classes, allowing developers to combine expert systems and Java applets and applications with minimal efforts. JESS is compatible with CLIPS in most parts, thus it can be used in reasoning systems in a way

[2] For a detailed review of teaching strategies & their implementation for the domain programming, see Liegle (1999).

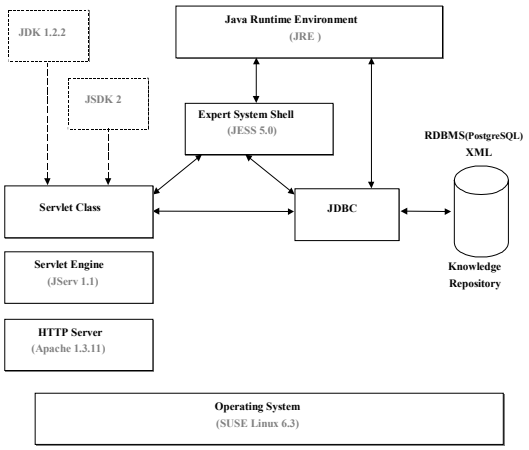

Figure 4: Physical Environment

that developer supply knowledge in the form of declarative rules[3].

The servlet engine JServ is used to provide the server side environment for the Java servlets. The entire system at this point is at a prototyping level of implementation, however, an XML interface will be provided at some point to allow the sharing of the knowledge repository with other intelligent applications.

For performance reasons, the fact-base of the expert system is created in two steps. The various fixed relationships (between lectures, between examples and solutions, and between learning styles and lectures) are only created when new knowledge is added to the system. These relationships are then stored as facts in a text file that can be either read by the servlet or hard-coded for performance reasons. When a student finishes a lecture and takes the post-test, the result of this test is added to the history file. Also, a fact-base is stored for each student representing the taken and passed/failed lectures. These student-facts are loaded into the expert system each time a new decision must be made in terms of what to teach next.

This user model represents the lesson structure of each course a user takes. Lessons are currently classified in one of five types: 1) "ready to be learned," 2) "mastery was demonstrated," 3) "mastery is assumed," 4) "prerequisites are missing," or 5) "learner tried and failed to master lecture." The status of a lecture is color-coded similar to the color-coding scheme of Weber and Specht (1997). Links to all lectures can be displayed in their respective colors to aid students under learner-control in their decision making. For example, a red color-coded link indicates that this particular lecture is not ready to be learned yet because some prerequisite lectures are not yet mastered. If a learner under learner control is given options of what to learn next, these links are annotated by the color codes, aiding the learner in the decision process on what to do next. The following JESS examples present 1) a representation of the lessons and 2) the color-coding scheme used to represent the status of a given lesson for a given learner.

Example: JESS representation of a lecture

```
(deftemplate lecture
  (slot name)    ; shortname of lecture
  (slot color))
```

Example: The status of the lesson "Variables" for a given user

```
(deffact lecture
  (name Variables); This user tried to
                  ; learn about Variables,
  (color orange)) ; but failed the test.
```

For each user, a file is created with a listing for each of the lectures. In the beginning, all lectures but the first one are marked "unlearned." This is an implementation of the widely used "overlay" model (Brusilovski, 1992), which requires a representation of the entire knowledge for each user.

Once a learner takes a test for a given lecture, the result of the test will impact the status of other lectures for which the current lecture is a prerequisite. For example, if the introduction to "loops" is mastered, then the learner can proceed with learning the details of a "for-loop". This would be indicated to the learner by changing the color code of annotated links from red (prerequisite knowledge for "for-loops" missing) to green (all prerequisites fulfilled). The update of the student model is again performed by rules of the expert system. Following are 1) the JESS template for a prerequisite relationship, 2) an example of an instance of such a relationship, and 3) an example of a JESS rule using this relationship.

Example: JESS representation of prerequisite relationship

```
(deftemplate prerequisite
  (slot A)   ; A is a prerequisite of B.
  (slot B))  ; A,B are lessons.
```

Example: One instance of a prerequisite relationship

```
(deffact prerequisite
  (A Variables) ; Understanding of
                ; Variables is required
  (B Input))    ; before Input can be
                ; explained.
```

After each learner interaction with the knowledge base, the student model is updated by the actual interaction (e.g. mastered lecture "introduction to loops"), then the rules of the expert system are applied to the student model to see what effect the change had, and the updated student model is saved for the next interaction.

[3] See http://herzberg.ca.sandia.gov/jess/

Example: One of the rules updating the user model after a change

```
(defrule myforward
  (topic (name ?first)
         (color ?firstcolor&black))
  (prereq (A ?first) (B ?second))
   ?x0 <- (topic (name ?second)
                 (color ?color&red))
  (not (exists
    (and (prereq (A ?otherfirst&~?first)
                 (B ?second)))
    (topic (name ?otherfirst)
           (color red|green)))
     =>
    (modify ?x0 (color green))  )
```

Additional rules are needed to aid in the decision process which version of a lecture should be given to what student. For example, a given lecture might have an abstract graphical and a concrete textual version. Which version should be given to a concrete but graphical oriented learner? We currently lack the data to make this decision, but plan on using our system to collect this information over time.

4. SUMMARY AND CONCLUSIONS

Personalized instruction can finally be delivered over the Internet through the use of adaptive web-based intelligent tutoring systems. While other systems have shown that web-based training is a feasible and economical way of delivering training, research has also shown that the mere presentation of information on the web does not qualify as instruction. The addition of testing tools can improve the learning, especially if students are sent back to review material that they have not mastered yet. And more, research has shown that if training material is presented to students according to their preferred learning style, learning improves. We have shown in this paper how training material can be organized in a way that allows an expert system shell to continuously monitor the progress of individual students, select the most appropriate next lecture, and present it in a way that best fits the preferred learning style of the learner. Once the implementation of our improved system is completed, we plan on conducting experiments that measure the amount of improved learning that takes place when personalizing instruction to individual students. We are currently limiting our efforts to the knowledge domain *Introductory Programming in C++*, but are planning on expanding the knowledge domain to incorporate other programming languages such as Visual Basic and Java and later math in order to further evaluate the impact of transfer knowledge on learning.

5. REFERENCES

Allinson, L. (1992) "Learning Styles and Computer--Based Learning Environments," *ICCAL Proceedings*, Wolfville, NS, Canada, pp. 61-73.

Bernstein, D.S. (1998) "WBT: Are We Really Teaching," *Inside Technology and Training* (2:2), pp. 15-17.

Bostrom, R.P., Olfman, L. and Sein, M.K. (1990) "The Importance of Learning Style in User Training," *MIS Quarterly* (14:2), pp. 101-119.

Brandt, D.S. (1997) "Constructivism: Teaching for Under of the Internet," *Communications of the ACM* (40:10), pp. 112-117.

Brusilovsky, P. (1998) "Adaptive Educational Systems on the World-Wide-Web: A review of Available Technologies," *4th International Conference on ITS Proceedings*, San Antonio, Texas.

Calhoun, J.F. (1975) "The Relation of Student Character-istics to Performance in a Personalized Course," *Educational Technology* (15:4), pp. 16-18.

Cole, K., Fischer, O. and Saltzman, P. (1997) "Just-in-Time Knowledge Delivery," *Communications of the ACM* (40:7), pp. 49-53.

Davidson, G.V., Savenye, W.C. and Orr, K.B. (1992) "How Do Learning Styles Relate to Performance in a Computer Applications Course?" *J. of Research on Computing in Education* (24:3), pp. 348-358.

Dede, C. (1996) "Emerging Technologies in Distance Education for business," *Journal of Education for Business* (71:4), pp. 197-204.

Dillon, N. (1997) "Internet-Based Training Passes Audit," *Computerworld* (31:44), pp. 47-48.

Egan, D. (1988) "Individual differences in human-computer interaction," In *Handbook of Human-Computer Interaction*, M. Helander (Ed.), Elsevier, Amsterdam, North-Holland, pp. 543-568.

Filipczak, B. (1996) "CBT: A Status Report," *Training* (33:11), p. 93.

Gibbons, P. L. (1997) "The Right Formula for Training," *Datamation* (43:9), pp. 96-101.

Gitomer, J. (1987) "Determining Who Will Benefit form Visual Interfaces," *Journal of Information Systems Management* (4:1), pp. 85-87.

Goforth, D. (1994) "Learner Control = Decision Making + Information," *J. Educational Computing Research* (11:1), pp. 1-26.

Hawkins, D.T. (1997) "WBT for Online Retrieval: Some Examples," *Online*, (9/10), pp. 73-74.

Herther, N.K. (1997) "Education Over the Web. Distance Learning and the Information Professional," *Online*, pp. 63-72.

Khan, B.H. (1997) "Web Based Instruction," Englewood Cliffs, New Jersey.

Knarr, J. (1996) "Effects of Cognitive Style and Mode of Information Presentation on the Usage of a Case Tool for Relational Data Modeling," unpublished Dissertation Draft, Kent State University.

Kolb, D. (1976) *Learning Style Inventory, Self-Scoring Test and Interpretation Booklet*, McBer and Company, Boston, MA.

Kruse, K. (1997) "Exploring Multimedia Internet-Based Training," *Training and Development* (51:3), pp. 55-56.

Lieberman, D.A. and Linn, M.C. (1991) "Learning to Learn Revisited" *Journal of Research on Computing in Education* (23:3), pp. 373-395.

Liegle, J.O. (1999) "Development and Evaluation of an Adaptive Web-Based Intelligent Tutoring System," Dissertation, Kent State University .

Liegle, J.O. and Madey, G.R. (1997) "Web Based Training," *Proceedings of the AIS 1997 Americas Conference*, Indianapolis, pp. 521-523.

Major, N. and Reichgelt, H. (1992) "COCA: A Shell for ITS," *ITS Proceedings*, Montreal, pp. 523-530.

Martinsons, M.G. and Schindler, F.R. (1995) "Organizational Visions for Technology Assimilation," *IEEE TA on Engineering Management* (42:1), pp. 9-18.

McGee, M.K. (1998) "Save On training," *Information Week*, June 22 , pp. 141-146.

McGrath, D. (1992) "Hypertext, CAI, Paper, or Program Control: Do Learners Benefit From Choices?," *Journal of Research on Computing in Education* (24:4), pp. 513-532.

Melara, G.E. (1996) "Investigating Learning Styles on Different Hypertext Environments," *J. Educational Computing Research* (14:4), pp. 313-328.

Merril, M.D. (1991) "Constructivism and Instructional Design," *Educational Technology* (31), pp. 45-53.

Merrill, M.D., Drake, L., Lacy, M.J., Pratt, J. and Group, I.R. (1996) "Reclaiming Instructional Design," *Educational Technology*, pp. 5-7.

Merrill, M.D. and Li, Z. (1989) "An Instructional Design Expert Sys.," *CBI Journal*(16:3), pp. 95-101.

Merrill, M.D., Li, Z. and Jones, M.K. (1992) "Instructional Transaction Shells," *Educational Technology*, pp. 5-26.

Moore, M.G. (1996) "Tips for the Manager Setting Up a Distance Education Program," *Journal of Distance Education* (10:1), pp. 1-5.

Murray, T. (1998) "Authoring Knowledge-Based Tutors: Tools for Content, Instructional Strategy, Student Model, and Interface Design," *The Journal of the Learning Sciences* (7:1), pp. 5-64.

Nelson, R.R., Whitener, E.M. and Philcox, H.H. (1995) "The Assessment of End-User Training Needs," *CACM* (38:7), pp. 27-39.

Niemiec, R.P., Sikorski, C. and Walberg, H.J. (1996) "Learner-Control Effects," *Journal of Educational Computing Research* (15:2), pp. 157-174.

Nkambou, R. and Gauthier, G. (1996) "Integrating WWW resources in an ITS," *Journal of Network and Computer Applications* (19), pp. 353-365.

Olfman, L. and Mandviwalla, M. (1994) "Conceptual Versus Procedural Software Training for Graphical User Interfaces: A Longitudinal Field Experiment," *MIS Quarterly* (18:4), pp. 405-426.

Reeves, T.C.(1993) "Pseudoscience in CBU," *J. of Computer Based Instruction* (20:2), pp. 39-46.

Rieman, J., Young, R. and Howes, A. (1996) "A dual-space model of iteratively deepening exploratory learning," *International Journal of Human-Computer Studies* (44:6), pp. 743-755.

Sadler-Smith, E. (1997) "'Learning Styles' and Instructional Design," *IETI* (33:4), pp. 185-193.

Schank, R.C., Korcuska, M. and Jona, M. (1995) "Multimedia Applications for Education and Training," *ACM Computing Surveys* (27:4), pp. 633-635.

Schnackenberg, H.L., Sullivan, H.J., Leader, L.F. and Jones, E.E.K. (1998) "Learner Preferences and Achievement Under Different Amounts of Leaner Practice," *ETRD* (46:2), pp. 5-15.

Schröder, O., Möbus, C. and Pitschke, K. (1995) "A Cognitive Model of Design Processes for Modeling Distributed Systems," *Proceedings of the Artificial Intelligence in Education*, Washington, DC, pp. 146-153.

Tan, S.-T. (1996) "Architecture of a generic instructional planner," *Journal of Network and Computer Applications* (19), pp. 265-274.

Tennyson, R.D. (1981) "Use of Adaptive Information for Advisement in Learning Concepts and Rules Using CAI," *American Educational Research Journal* (18:4), pp. 425-438.

Wells, J.B., Layne, B.H. and Flowers, C.P. (1993) "Assessment of CBI Use in Criminal Justice Education Programs," *J. Educational Technology Systems* (22:1), pp. 57-67.

Young, J.D. (1996) "The Effect of Self-Regulated Learning Strategies on Performance in Learner Controlled CBI," *ETR&D* (44:2), pp. 17-27.