Das Virtuelle Informatik-Labor als Teil des Lernraums Virtuelle Universität der FU Hagen

Rainer Lütticke und Hermann Helbig

FernUniversität Hagen, Fachbereich Informatik
Intelligente Informations- und Kommunikationssysteme
Universitätsstr. 1
58084 Hagen, Deutschland
rainer.luetticke@fernuni-hagen.de

Abstract: Durch das Internet haben sich Chancen für eine qualitativ verbesserte Lehre, gerade in der Fernlehre, ergeben. Damit den Studierenden der FernUniversität Hagen ein Portal zu den Lehrangeboten, die über das Internet erreichbar sind, geboten wird, wurde der Lernraum Virtuelle Universität eingerichtet. Ein Bestandteil dieses Lernraums ist das virtuelle Informatik-Labor. In dem als Client-Server aufgebauten System werden den Studierenden Aufgaben mit verschiedenen Inhalten angeboten. Die Studierenden setzen zur Lösung der Aufgaben Software-Werkzeuge ein und werden durch eine interaktive tutorielle Komponente zur Korrektur der Lösungen unterstützt.

1 Einleitung

Das betreute Fernstudium bietet ein Höchstmaß an Unabhängigkeit und Flexibilität und eignet sich deshalb für Studieninteressenten, die sich aufgrund ihrer persönlichen Situation an keiner Präsenzhochschule einschreiben könnten. Es ist eine ideale Studienform für das lebensbegleitende Lernen, die Weiterbildung neben dem Beruf. Ungünstig kann sich teilweise fehlende Präsenz der Studierenden an einer Universität und daraus resultierende Schwierigkeit des Selbststudiums auswirken. Die zeit- und kostenintensiven Anreisen zu Präsenzphasen sind zudem ein Nachteil. Mit der Leitentscheidung im Jahre 1999 zum flächendeckenden Einsatz neuer Medien und zur Entwicklung des Lernraums Virtuelle Universität (LVU[1]) hat die FernUniversität (FU) Hagen aber auf die Anforderungen der Studierenden nach weiterer Erhöhung der Flexibilität und neuen Modellen zur Qualitätssteigerung in der Fernlehre reagiert. Mit dem Konzept LVU macht die FU Hagen alle Funktionen einer Universität über das weltweite Netz verfügbar: Lehre, Forschung, Verwaltung und die damit verbundenen Informations-, Kommunikations- und Kooperationsmöglichkeiten werden über die informationstechnische Infrastruktur des Internets angeboten. Studierende können so zunehmend Studienmaterialien sowie ergänzende Angebote über das Internet nutzen. Die virtuelle Mobilität hebt die Begrenzungen von Zeit und

[1] http://www.fernuni-hagen.de/LVU/index.html

Raum auf, so dass Präsenzphasen reduziert werden können. Der Online-Austausch mit anderen Teilnehmern und Betreuern garantiert die fachliche und soziale Einbindung und löst so die Isolation der Studierenden auf. Moderne Lerntechnologien und neue Lernangebote erleichtern den Übergang vom Lehren zum Lernen und eröffnen Möglichkeiten, sich individualisiert und selbst gesteuert Wissen und Kenntnisse anzueignen: Modularisierte Lehrmaterialien, individualisiert verwendbare Übungselemente, aktive Informationsvermittlung, die Erschließung der internationalen Fachwelt und wissenschaftlicher Diskussionen sowie des konkreten Praxisbezugs erlauben es, persönliche Lernpfade zu verfolgen. Durch Einrichtung des LVU-Büros im Rektorbüros werden alle Aktivitäten im Bereich der Virtuellen Universität wirksam unterstützt und koordiniert.

Durch die Umsetzung dieser strategischen Vorgabe (auch schon im Vorfeld der Entscheidung durch Forschungprojekte) verfügt der Fachbereich Informatik der FU Hagen über mehrjährige Erfahrungen in der Internet-basierten Lehre. Insbesondere im Bereich der Gestaltung und des Anbietens multimedialer Lehrinhalte durch die virtuelle Universität (VU) ([SM98]) und der Unterstützung des online-Übungsbetriebes (WebAssign; [BHSV99]) wurden im Fachbereich Informatik bereits umfangreiche Arbeiten geleistet und Systeme entwickelt, die bereits erfolgreich ihren Regelbetrieb aufgenommen haben. Ein spezielles Multimedia-Förderprogramm, das seit 2000 an der FU Hagen besteht, und weitere extern eingeworbene Forschungsprojekte auf dem Gebiet des E-Learnings haben zu weiteren Innovationen geführt. Anfänglich durch das Förderprogramm 2001 initiiert und nun durch das BMBF gefördert, ist das virtuellen Informatik-Labor (VILAB[2]) Teil des LVU und wird seit dem SS 2002 im Lehrbetrieb an der FU erfolgreich eingesetzt.

VILAB schafft eine virtuelle Laborumgebung, die es den Studierenden (ähnlich wie in anderen Naturwissenschaften in realen Labors) erlaubt, ihre in den Lehrveranstaltungen erworbenen theoretischen Kenntnisse in praktischen Versuchen zu erproben. Dazu werden den Studierenden in VILAB komplexe Softwarekomponenten angeboten, mit denen sie (Simulations-)Experimente sowie Eigen- und Weiterentwicklungen von grundlegenden Verfahren der Informatik durchführen können. Dadurch ergeben sich für die Studierenden mehrere Vorteile: Sie sparen die zeit- und kostenaufwendige Anreise für reale Praktika, sie können sich die Lernzeiten frei einteilen, der Transfer von theoretischem Wissen in praktische Anwendung wird gefördert ([MRGC95], [Hay97], [B$^+$00]) und ein zusätzlicher Lernerfolg durch den motivationsfördernden Einsatz interaktiver Elemente stellt sich ein.

2 Architektur

VILAB ist nach einem Client-Server-Modell aufgebaut, das sich im Wesentlichen aus einem in der Universität betriebenen Lehrserver und den Windows- oder Linux-basierten PCs als Client bei den Studierenden zusammensetzt. Server-seitig besteht das System aus einem Kern auf Linux-Basis, der die Laborinhalte speichert und verwaltet und über ein

[2]VILAB ist Teil des medin-Projekts (Multimediales Fernstudium in der Medizinischen Informatik), welches durch das BMBF innerhalb des Programms “Neue Medien in der Bildung” gefördert wird.

Navigationstool (im Lisp-Dialekt Scheme geschrieben) das Interface für den Nutzer liefert. Der Systemkern verfügt außerdem über eine tutorielle Komponente (Java), die interaktiv Hilfestellungen zu den gerade bearbeiteten Aufgaben abgeben kann. Die genauen inhaltlich-didaktischen Führungsalgorithmen und Kontrollmechanismen für eine konkrete Laborstation werden allerdings in letzterer zur Verfügung gestellt ([LGH02]). Der Plattform-Kern verwaltet auch die in das Lernsystem integrierten Werkzeugen und stellt die Kommunikations-Infrastruktur zu den übrigen Plattform-Komponenten sowie zu WebAssign bereit. WebAssign ist dabei so in VILAB integriert, dass der Nutzer nicht merkt, bei welcher Aufgaben er sich im WebAssign-System befindet.

Der Zugang zu VILAB wird über dezentrale Benutzer-Clients vorgenommen, damit die Studierenden selbstständig von überall mit dem System via Internet arbeiten können. Die Studierenden gelangen dabei durch ein Remote-login zum Navigationstool auf dem VILAB-Server. Dieser ist so konfiguriert, dass nach dem Login eine Instanz des Labors gestartet und automatisch das Sende-/Empfangssystem der tutoriellen Komponente aktiviert wird. Dadurch kommen die Studierenden weder gewollt noch ungewollt mit dem unterliegenden Linux-System in Berührung. Zur Anzeige der Laborinhalte öffnet der Nutzer bei sich auf dem Rechner einen Browser mit einer ihm dauerhaft zugeteilten individuellen URL. Über das Navigationstool sind Software-Werkzeuge, Seiten aus dem WWW sowie lokale PDF- und HTML-Dokumente ansteuerbar (Guided Tour[3]). Die Prozesse der Software-Werkzuge werden dabei vom System überwacht. Aktionen im Navigationstool, die zur Anzeige von Dokumente führen sollen, resultieren in einer automatischen Aktualisierung der Seite im Browser. Dies wird dadurch realisiert, dass die zentral im System abgelegten lokalen Dateien in das individuelle Directory des Nutzers in VILAB geladen und zur Anzeige auf die Datei der entsprechenden URL des Nutzers kopiert werden. Durch PHP-Skripte erkennt der Browser eine Veränderung des Inhalts unter der unveränderten URL und stößt ein automatisches Reload der Seite an. Auf diese Weise ist es möglich, dass der Server den Client-seitigen Browser steuert. Externe Dokumente aus dem Internet werden in einen Frame eingebettet, so dass auch dort übergeordnete PHP-Skripte den Inhalt des Browsers unter der URL des Nutzers kontrollieren können. Dadurch dass jeder Nutzer innerhalb des Systems individuelle Ports und Directories besitzt, können Datenanzeige und das Speichern von Dateien gemanagt werden und mehrere Studierende zur gleichen Zeit im Labor arbeiten.

3 Laborstationen und Lehrinhalte

Die Lehrinhalte sind in verschiedene Laborsäulen mit mehreren Aufgaben unterteilt, die teils aufeinander aufbauen bzw. aufeinander Bezug nehmen und teils eigenständig sind. Das Spektrum reicht von einfachen Aufgaben in der Programmierung (eingebettet sind alle grundlegenden Paradigmen der Programmierung: prozedural (C), objektorientiert (Java), funktional (Scheme) und logisch (Prolog)) über Neuronale Netze bis zu Übungen zu inferenzbasierten Methoden aus dem Bereich der automatischen Sprachverarbeitung mit Anbindung relationaler Datenbanken (SQL). Insbesondere für die semantische Repräsen-

[3] http://pi7.fernuni-hagen.de/vilab/tour/

tation natürlichsprachlicher Informationen wurde ein grafisches Werkzeug zur Wissensrepräsentation und -verarbeitung (MWR [Gnö00]) in VILAB integriert.

Bei der Konzeption des Kernsystems wurde ein besonderes Augenmerk darauf gerichtet, die modulare Erweiterbarkeit des virtuellen Labors zu unterstützen, damit aufbauend auf der entwickelten Plattform zukünftig weitere Laborstationen mit neuen Lehrinhalten und bestehenden Softwarewerkzeugen realisiert und in VILAB integriert werden können. Mit Lehrstühlen an der FernUniversität Hagen, der TU München, der Medizinischen Universität zu Lübeck und der RWTH Aachen wird diesbezüglich bereits kooperiert. Zu diesem Zweck sind spezielle Datei-Schablonen (in HTML und Scheme) für neue Aufgaben und Lehrstationen bereitgestellt, die von den Autoren nur noch ausgefüllt werden müssen. Jedoch wird zur Zeit ein Autorentool entwickelt, um die Einrichtung neuer Laborstationen zu vereinfachen.

4 Benutzer-Clients

Der Zugang der Studierenden zu dem System wird über dezentrale Benutzer-Clients vorgenommen, damit die Studierenden selbständig von zu Hause mit dem System arbeiten können. Die Herstellung der Verbindung zwischen dem Lehrserver und den Clients geschieht über das Internet. Um nahezu allen Studierenden den Zugang zum Labor technisch zu ermöglichen, muss der Benutzer-Client auf x86-Rechnern mit einer durchschnittlichen Hardware-Ausstattung und den Betriebssystemen Windows 9x, Linux oder kompatiblen Versionen lauffähig sein. Client-seitig wird vorzugsweise nur frei verfügbare Software eingesetzt, die leicht beschaffbar und installierbar ist. Das Remote-login ist für Studierende, die bereits Linux einsetzen, trivial. Windows-Benutzer müssen zusätzlich die Open-Source-Pakete *cygwin* und *XFree86 4.x* installieren, um durch die so bereitgestellte entsprechende Unix-Funktionalität den Zugang herstellen zu können. Zu diesem Zweck wurde eine CD-Distribution mit den o.g. Softwarepaketen speziell für VILAB konzipiert und zusammengestellt, die bei Bedarf an Studierende ausgegeben wird. Der Nachteil einer möglichen notwendigen Software-Installation wird aber durch die Vorteile eines Remote-logins mehr als aufgewogen. Durch den Verzicht auf die Implementierung der Benutzeroberfläche innerhalb eines Internetbrowsers ergeben sich umfangreichere technische und inhaltliche Gestaltungsmöglichkeiten für das Labor. Dadurch gewährleistet der gewählte Zugang zu VILAB eine wesentlich bessere Performanz und Darstellungsqualität als beispielsweise ein ausschließlicher Einsatz einer Browser-Umgebung, bei dem die Ein- und Ausgabeschnittstellen der Laborwerkzeuge auf ein Java-Applet umgeleitet werden müssten. Da der Nutzer während seiner Arbeit im Labor permanent online sein muss, können sich die Studierenden alle im Labor aufrufbaren Dokumente auch über eine gewöhnliche URL zur offline-Nutzung herunterladen.

5 Interaktive tutorielle Komponente

Die Steuerung und Überwachung der Laborarbeit wird durch die Schnittstelle zu der interaktiven tutoriellen Komponente mit seinem Sende/Empfangssystems (Java) vorgenommen. Zu den Hauptaktivitäten, die über diese Schnittstelle abgewickelt werden, gehören die folgenden Aktionen:

- Bereitstellung der Aufgaben in verschiedenen Formaten (HTML, PDF)
- Einsendung einer (Teil-)Lösung verbunden mit der Aufforderung an die Tutor-Komponente, eine Bewertung oder eine Zwischenstandsmeldung abzugeben.
- Übermittlung eines Fehlerzustandes durch ein Laborwerkzeug, zu der erklärende Informationen angezeigt werden sollen.

Zur konkreten Ausgestaltung der Tutor-Schnittstelle existieren ein internes und ein externes Tutormodell, die beide eine intelligente Analyse der studentischen Lösung nach Aufforderung durchführen ([Bru99]), aber einen unterschiedlichen Grad an Eigenständigkeit der Lehrmodule voraussetzen. Das externe Modell bietet zusätzlich die Möglichkeit einer interaktiven Problemlösungsunterstützung ([Bru99]).

Der **externe Modus** ist für Laborkomponenten gedacht, die weitgehend autark mit dem Benutzer interagieren. Ein Beispiel für eine derartige Komponente ist das grafische Werkzeug zur Wissensrepräsentation (s. Kap. 3), das über eine eigene Benutzeroberfläche zur Anzeige und Manipulation der Wissensbasis verfügt. In diesem Fall findet die eigentliche Interaktion mit dem Benutzer und die Analyse von Lösungen durch die Komponente der Lernumgebung selbst statt. Interaktive Problemunterstützungen können hier so ausgestaltet sein, dass gewisse Aktionen im Software-Werkzeug, die als falsch erkannt werden, sofort durch erläuternde Fehlertexte in Fenstern oder durch farbliche Änderungen im Werkzeug angezeigt werden (z.B. rote Farbe für falsche Elemente). Zur Kommunikation mit den Laborkomponenten dient in Fall der Aufforderung zur Analyse der kompletten Lösung eine Socket-Verbindung; da Socket-Schnittstellen unter fast allen gängigen Programmiersprachen vorhanden sind, ergibt sich so eine optimale Unterstützung zur Einbindung bereits vorhandener Softwarekomponenten. Ist die Analyse abgeschlossen, sendet das Software-Werkzeug einen Fehlercode und eine dynamisch erzeugte Datei mit Ergebnissen der Analyse an den Tutor. Der Tutor selbst wird mit Hilfe einer Ressourcendatei vorkonfiguriert, in der für Fehlercodes die vom Tutor zu ergreifenden Aktionen (z.B. Fehlertexte anzeigen) angegeben sind. Aus dem Fehlertext und der Datei mit den Ergebnissen der Fehleranalyse wird nun mittels PHP, Javascript und HTML dynamisch die Feedback-Datei erzeugt, die individuell auf den Fehler des Nutzer abgestimmt ist. Diese wird in das individuelle VILAB-Directory des Nutzers geladen und wie oben beschrieben im Browser angezeigt.

Im **internen Modus** hat der Tutor eine stärkere Kontrolle über den Ablauf der Softwarekomponenten im Labor. Dieser Modus wird typischerweise bei der Einbindung kommandozeilenorientierter Komponenten wie z.B. von Programmiersprachen-Übersetzern oder SQL angewendet. In diesem Fall behält der Tutor nach dem Starten die Kontrolle über den

Prozess, indem die Standard-Eingabe- und Ausgabe-Datenströme des Prozesses permanent überwacht werden. Da die Interpretation der Standard-Ausgabe hochgradig von der Syntax der jeweiligen Softwarekomponente abhängig ist, kann die Reaktion des Tutors in diesem Fall nicht durch einen einfachen Ressourceneintrag festgelegt werden, sondern es muss durch den Autor der betreffenden Lehrstation eine geeignete Java-Klasse innerhalb des Tutors bereitgestellt werden. Diese Java-Klasse stellt innerhalb des WebAssign-Systems einen sogenannten "Korrektur-Server" dar. Dieser kann von den VILAB-Nutzern (via CORBA) zur Korrekturanfrage oder Speicherung einer Aufgabenlösung angesprochen werden. Durch die Verarbeitung der Eingaben und nachfolgend ausgelöste Prozesse (z.B. Compilieren von Programmen) werden vom Tutor im Labor entsprechende Reaktionen darauf veranlasst (z.B. Fehlertexte, s.o.).

Inhaltlich besteht das Feedback einer Fehleranalyse aus den drei Teilen Fehlerdiagnose, Fehlerkorrektur und Leistungsbewertung, die jeweils nach Aspekten für motivationales Feedback gestaltet sind ([Mus00]). In der Fehlerdiagnose wird nur kurz auf den Grad der Richtigkeit der Lösung eingegangen, während die Fehlerkorrektur alle Hinweise und Hilfen enthält. Diese können in den einfachsten Fällen kommentierte Musterlösungen und das Aufzeigen von Syntaxfehlern sein. Durch komplexere Test- und Inferenzverfahren aus der künstlichen Intelligenz können aber auch einzelne Fehler herausgefunden und abhängig von den Fehlern passende Links zu Lehrtexten angegeben werden. Des Weiteren sind auch Hilfen in Form von Beispielen und Wegen zur Lösungsfindung möglich. Entsprechend der Analyse wird dann eine kurze Leistungsbewertung abgegeben, die besonders für die Vermittlung motivationalen Aspekte des Feedbacks genutzt wird.

Nach dem Grad und der Funktionsweise an Interaktivität und der Fehlerdiagnose/-korrektur beim Aufgabenlösen kann VILAB somit als eine "Intelligent Learning Environment" betrachtet werden (nach der Definition von Dillenbourg et al. ([DHM+93]) in [Bui99]).

6 Curriculare Einbettung und erste Erfahrungen

VILAB wird in verschiedenen Bereichen an der FU Hagen in der Informatik-Lehre eingesetzt. In einem Seminar dient es als Hilfestellung und Vertiefung des jeweiligen Seminararbeitsthemas, so dass mit zwei Präsenzphasen VILAB für Blended Learning eingesetzt wird. Da zwar die Belegung eines Seminars (Bachelor) bzw. von zwei Seminaren (Diplom) zum Pflichtprogramm eines Studierenden an der FU Hagen gehört, aber mindestens sechs Seminare pro Semester zur Auswahl stehen, kann der Studierende aussuchen, welchen multimedialen Grad sein Seminar haben soll: klassisch, blended learning oder komplett virtuell mit dem LVU als Seminarort. In einem Praktikum konnten Studierende optional Teillösungen von Programmen durch die automatische Tutor-Komponente überprüfen lassen. Auch hier gehört mindestens ein Praktikum zum Pflichtprogramm und auch hier gibt es Wahlmöglichkeiten zwischen verschiedenen Praktika (zwei pro Semester, von Semester zu Semester unterschiedlich), die den LVU unterschiedlich stark nutzen. Des Weiteren wird das Labor in einem Kurs von den Studierenden zur Unterstützung bei der Lösung von komplexen Übungsaufgaben und für Selbsttestaufgaben eingesetzt. Außerdem wird VILAB ungebunden von Lehrveranstaltungen als Experimentierfeld und zur Prüfungsvor-

bereitung verwendet und die Softwarewerkzeuge kommen im Labor bei Diplomarbeiten zum Einsatz.

Eine Evaluation zeigt positive Ergebnisse. Eine Befragung der Studierenden ergab, dass die Windows-Nutzer (85%) die zusätzliche Software problemlos installieren konnten. Die Kommunikation aller Betriebssysteme (Windows-Versionen und Unix-Derivate) mit der Plattform funktionierte einwandfrei. Die Reaktionszeiten der Software-Tools und des PHP-gesteuerten Client-seitigen Browser waren sehr gut (selbst mit 56k-Modem). Die Navigationsmöglichkeiten wurden als gut bewertet, ebenfalls auch die Struktur der Aufgabenstellungen. Besonders die tutorielle Komponente hat durch das schnelle Feedback geholfen und motiviert. Weitere positive Auspekte waren: abwechslungsreiche Aufgabenstellungen, intessantes Arbeiten mit komplexen interaktiven Werkzeugen und die Möglichkeit, sich Lösungen "spielerisch" durch try-and-error zu erarbeiten. Einigen Studierenden fehlten kollaborative Elemente, jedoch war den meisten Studierenden die gute Erreichbarkeit des Betreuers wichtiger. Objektiv zeigte sich, dass die Aufgabenlösungen durch die tutorielle Unterstützung überwiegend sehr gut waren und viel besser als die Lösungen ohne tutorielle Unterstützung. Außerdem zeugten die Seminarvorträge davon, dass die Vertiefungsmöglichkeiten zu einem Thema zu einem besseren Verständnis führten. Äußerst beachtenswert ist die Tatsache, dass die Quote der erfolgreichen Kursteilnehmer deutlich gestiegen ist (95% gegenüber sonst $\sim$ 50%) und 25% der Seminarteilnehmer nun an einem Diplomarbeit aus dem Themenberich von VILAB arbeiten.

Die optionale Möglichkeit der Nutzung im Praktikum wurde hingegen wenig angenommen, allerdings gerade von denen, die das Praktikum knapp erfolgreich abgeschlossen haben. Die angebotene Hilfe wurde also nur als "letzte Rettung" angesehen. Jedoch auch diesen Zweck erfüllte die Lernumgebung, wie die Ergebnisse zeigen. Allerdings ziehen wir aus den positiveren Ergebnissen der gezwungenen gegenüber der freiwilligen Nutzung von VILAB die Schlussfolgerung, dass Studierenden nicht die Wahl gelassen werden sollte.

7 Ausblick

Die Weiterentwicklung des Labors zielt vor allem auf die Verfeinerung der tutoriellen Komponente sowie deren Einsatz in neuen Laborsäulen, so dass sich die Einsatzmöglichkeiten von VILAB verbreitern. Dazu bestehen neben den schon erwähnten Kooperationen (s. Kap. 3) eine weitere mit der Universität Erfurt, um neuste Ergebnisse der Mediendidaktik in VILAB anwenden zu können, denn im Bereich des intelligenten Feedbacks gibt es noch ein großes Potenzial für VILAB. Besonders durch die zukünftige Integration eines Studentenmodells werden sich noch bessere Reaktionsmöglichkeiten für die tutorielle Komponente ergeben. Außerdem sollen kooperative Software-Werkzeuge eingebunden werden, damit das psychologische Bedürfnis nach sozialer Eingebundenheit besser befriedigt werden kann ([DR85]), denn bisher sind nur Newsgroup (unter den Studierenden) und E-Mail-Verkehr (zwischen Lehrendem und Lerner) als Kommunikation außerhalb des Dialoges mit dem Lernsystem im Einsatz.

Um die Nachhaltigkeit von VILAB sicherzustellen, wird das Labor nach der Entwicklungsphase von einer zentralen Stelle im Fachbereich Informatik administriert werden. Dieses Schrittweise Vorgehen wurde auch beim WebAssign-System angewendet und hat sich bewährt.

Insgesamt ist der LVU und VILAB als Teil und als Beispiel davon sehr gut von den Studierenden angenommen worden, so dass der eingeschlagene Weg von der FU Hagen bzgl. LVU und von unserem Lehrgebiet bzgl. VILAB mit großer Zuversicht weiterverfolgt wird.

Literaturverzeichnis

[B+00] S.-P. Ballstaedt et al. *Planung, Entwicklung, Durchführung von Fernstudienangeboten*. Deutsches Institut für Fernstudienforschung an der Universität Tübingen, 2000.

[BHSV99] J. Brunsmann, A. Homrighausen, H.-W. Six, and J. Voss. Assignments in a Virtual University - The WebAssign-System. In *Proc. 19th World Conference on Open Learning and Distance Education*, Vienna, Austria, 1999.

[Bru99] P. Brusilovsky. Adaptive and Intelligent Technologies for Web-based Education. In *Künstliche Intelligenz, 13(4)*, pages 19–25, 1999.

[Bui99] C. Buiu. *Artificial Intelligence in Education - State of the Art and Perspectives*. Zentrales Institut für Fernstudienforschung an der FernUniverisität Hagen, 1999.

[DHM+93] P. Dillenbourg, M. Hilario, P. Mendelsohn, D. Schneider, and B. Borcic. *Intelligent Leeaning Environments. Report from the project "Les systemes exploraturs intelligents"*. TECFA, Universität Genf, 1993.

[DR85] E.L. Deci and R.M. Ryan. *Intrinsic motivation and self-determination in human behavior.* Plenum, New York, 1985.

[Gnö00] C. Gnörlich. *MultiNet/WR: A Knowledge Engineering Toolkit for Natural Language Information. Technical-Report 278*. FernUniversität Hagen, 2000.

[Hay97] W.J. Haynie. Effects of Anticipation of Tests on Delayed Retention Learning. *Journal of Technology Education*, 9, 1997.

[LGH02] R. Lütticke, C. Gnörlich, and H. Helbig. VILAB - A Virtual Electronic Laboratory for Applied Computer Science. In *Proceedings of the Conference Networked Learning in a Global Environment*, pages 135 + CD–ROM, Canada/The Netherlands, 2002. ICSC Academic Press.

[MRGC95] G.R. Morrison, S. M. Ross, M. Gopalakrishnan, and J. Casey. The Effects of Feedback and Incentives on Achievement in Computer-Based Instruction. *Contemporary Educational Psychology*, 20:32–50, 1995.

[Mus00] J. Musch. Die Gestaltung von Feedback in computergestützten Lernumgebungen: Modelle und Befunde. *Zeitschrift für Pädagogische Psychologie*, 13:148–160, 2000.

[SM98] G. Schlageter and S. Mittrach. Virtuelle Universität. In *Informatik Forschung und Entwicklung, Volume 13*, pages 159–162, 1998.