Integration des Virtuellen Prüfungssystems ViPS in die Lehr-/Lernplattform Stud.IP

Philipp Hügelmeyer[1]
Universität Osnabrück
Zentrum virtUOS
49069 Osnabrück

Robert Mertens [2]
Universität Osnabrück
Zentrum virtUOS
49069 Osnabrück

Martin Schröder[3]
Universität Osnabrück
Zentrum virtUOS
49069 Osnabrück

Helmar Gust[4]
Universität Osnabrück
Institut für Kognitionswissenschaft
49069 Osnabrück

Zusammenfassung

Das virtuelle Prüfungssystem ViPS und seine Vorgänger MVC und VC-Prolog-Tutor werden seit mehreren Jahren an der Universität Osnabrück entwickelt. Zur Vereinfachung von Wartbarkeit und Distribution sowie Organisation und Verwaltung wurde das System in die seit 2003 hochschulweit an der Universität Osnabrück eingeführte Lernplattform Stud.IP integriert. Der Beitrag gibt zunächst einen Überblick über die Integration von ViPS in Stud.IP und schildert anschließend aktuelle Erweiterungen des ViPS.

1. Einleitung

An der Universität Osnabrück wird seit mehren Jahren ein tutorielles System zur automatisierten Auswertung von Übungs- und Klausuraufgaben eingesetzt, dass ab 1997 im Rahmen des Projektes Virtueller Campus entwickelt wurde. Das System wurde während dieser Zeit in drei verschiedenen Versionen eingesetzt um jeweils aktuellen Anforderungen gerecht zu werden. Mit der hochschulweiten Einführung der Lehr-/Lernplattform Stud.IP an der Universität Osnabrück im Sommer 2003 sind die Weichen zur Entwicklung der aktuellen Version des Systems gestellt worden.

Durch die Integration des virtuellen Prüfungssystems (ViPS) in die Lehr-/Lernplattform konnten nicht nur organisatorische Prozesse beim Einsatz des Prüfungssystems vereinfacht werden. Auch technische Hürden konnten aus dem Weg geräumt werden.

Da das ViPS als Stud.IP-Modul realisiert ist, kann es kursspezifisch ein- bzw. ausgeblendet und verwaltet werden. Aus Dozentensicht existiert damit zu jedem Kurs, in dem das ViPS aktiviert wurde, eine unabhängige Instanz des Systems. Durch diese Art der Kopplung an die Lernplattform können Aufgaben und Klausuren

[1] philipp.huegelmeyer@uni-osnabrueck.de
[2] robert.mertens@uni-osnabrueck.de
[3] martin.schroeder@uni-osnabrueck.de
[4] helmar.gust@uni-osnabrueck.de

kursspezifisch verwaltet werden. Ein separater Import von Teilnehmerinformationen entfällt, da die entsprechenden Daten direkt aus der Lehr-/Lernplattform übernommen werden.

Ein weiterer Vorteil besteht in der vereinfachten Wartbarkeit und Distribution des ViPS. Das ViPS kann ohne weiteres in aktuelle Stud.IP-Versionen integriert werden, indem die ViPS-Dateien in das bestehende Stud.IP kopiert werden und drei Dateien in der Stud.IP-Installation modifiziert, bzw. ausgetauscht werden. Darüber hinaus erfordert das ViPS den Betrieb einer zusätzlichen Datenbank, um Inkonsistenzen mit der bereits bestehenden Stud.IP Datenbank bei Versionsänderungen zu vermeiden.

Änderungen im Vergleich zu seinem Vorgänger, dem MVC (Minimal Virtual Campus) hat das ViPS in erster Linie in den Punkten Look and Feel und aufgrund einer Anpassung an das Usabilitykonzept der Lernplattform erfahren. Darüber hinaus sind neuere Techniken zur automatischen Korrektur größerer Texte und weitere Aufgabentypen wie mathematische Aufgaben (mit der Matlab ähnlichen Berechnungssprache Octave) hinzugekommen. Auch ist durch die Integration der Stud.IP Formatierung die direkte Eingabe von LaTeX, die Textformatierung und das Einbinden von Bildern und Verweisen sowohl in die Aufgabenstellung, als auch in Lösungen, Kommentaren etc. hinzugekommen.

Zur Auswertung und Benotung ganzer Kurse wird derzeit ein fortgeschrittenes Bewertungsverfahren entwickelt. Ein besonderer Fokus liegt dabei auf dem Umgang mit dem Ausreißen einzelner Ergebnisse.

Ziel dieses Papers ist neben der Darstellung der umrissenen Neuerungen im ViPS die Diskussion von Problemen und Erfahrungen im Zusammenhang mit der Integration des ViPS in eine bestehende Lehr-/Lernplattform.

2. Stud.IP

Stud.IP ist ein webbasiertes Open-Source-Lernmanagementsystem zur Unterstützung von Präsenzlehrveranstaltungen, das an der Universität Göttingen entwickelt wurde [Stu05]. Es ist damit an die Gegebenheiten der deutschen Hochschullandschaft angepasst und erfüllt so eines der wichtigsten Kriterien zum hochschulweiten Einsatz an der Universität Osnabrück [TR03]. Seit Sommer 2003 wird Stud.IP an der Universität Osnabrück als hochschulweite Lernplattform eingesetzt [The04]. Dank einer Anbindung der Authentifizierung der Nutzer der Lernplattform an den LDAP-Server des Rechenzentrums kann jeder Dozent und Studierende die Lernplattform ohne erhöhten Verwaltungsaufwand nutzen.

Aufgrund des Open-Source-Characters der Lernplattform ist es möglich, individuelle Anpassungen vorzunehmen und auch eigene Module in die Lernplattform zu integrieren. So ist beispielsweise durch Tobias Thelen vom Zentrum virtUOS ein WikiWikiWeb für Stud.IP entwickelt worden [WB04]. Im Zusammenhang mit dem hochschulweiten Einsatz der Lernplattform ist am Zentrum virtUOS darüber hinaus weiteres Wissen zur Erweiterung und Modifikation der Lernplattform gesammelt worden.

3. ViPS

Die entwicklungsgeschichtlichen Wurzeln des ViPS reichen bis in das Jahr 1997 zurück. Im Rahmen des Projektes Virtueller Campus wurde seiner Zeit zunächst mit der Entwicklung des VC-Prolog-Tutors begonnen [Pey99], [Wag00]. Später wurde ein Lisp-Modul für das System entwickelt [Pey00] und beide Module wurden in mehreren Lehrveranstaltungen zunehmend erfolgreich eingesetzt.

Nachfolger des Prologtutors war zunächst der MVC, der einige technische Neuerungen mit sich brachte und sowohl Wartung als auch Installation vereinfachte [Gus03] und an der Universität Osnabrück in einer Reihe von Lehrveranstaltungen eingesetzt wurde [HM04]. Da auch der MVC noch ein eigenständiges System war, wurden im Einsatz oft Kurs- und Studierendendaten doppelt verwaltet. Dies führte zu einem erheblichen Mehraufwand, der gerade Erstbenutzer auf Seiten der Dozenten oft abschreckte und auch der Einstellung der Studierenden zur Nutzung des Systems schadete. Auch die Oberfläche und die Bedienphilosophie waren für Nutzer oft gewöhnungsbedürftig. Hinzu kam die Tatsache, dass das System eine eigene Datenbank und Infrastruktur benötigte.

Um diesen Problemen zu begegnen, wurde das Virtuelle PrüfungsSystem ViPS als integrierter Bestandteil der Osnabrücker Stud.IP-Installation entworfen.

4. Automatische Auswertung von Aufgaben

Eines der interessantesten Features des virtuellen Prüfungssystems ist ohne Zweifel die automatische Korrektur von Übungsaufgaben. Während dieses bei Multiple Choice oder Zuordnungsaufgaben noch relativ einfach ist, sind für verschiedene andere Aufgabentypen andere Korrekturideen entwickelt worden, die an dieser Stelle vorgestellt werden.

4.1. Kurze Textlösungen

Bei einer kurzen Textlösung handelt es sich um eine Aufgabe, auf die eine Antwort von bis zu ein paar Wörtern gegeben werden kann. Der Autor der Aufgabe kann dazu mehrere Lösungen angeben, und sagen, ob die jeweilige Lösung richtig oder falsch ist. Das kann zum Beispiel für die Eingabe von Synonymen genutzt werden.

Je nachdem, wofür Textaufgaben einsetzen werden sollen, werden an die Auswertung unterschiedliche Anforderungen gestellt. Deswegen wurden drei verschiedene Bewertungsalgorithmen integriert. Ganz klassisch wird gar kein Ähnlichkeitsmaß verwendet, sondern nur geprüft, ob der Text identisch mit einer der angegebenen Lösungen ist. In diesem Fall wird die Lösung als korrekt gewertet. Des Weiteren werden zwei klassische Algorithmen zur Bestimmung von Textähnlichkeiten benutzt, wie sie in [Knu98] beschrieben sind. Soundex bestimmt die phonetische Ähnlichkeit von zwei Ausdrücken, indem es für beide jeweils eine verkürzte phonetische Darstellung berechnet. Levenshtein hingegen gibt für zwei Ausdrücke die textliche Ähnlichkeit an.

4.2. Lange Texte

Die automatische Korrektur von längeren Textaufgaben ist ungleich schwieriger, da hier nicht von der gleichen Wortwahl und Textlänge ausgegangen werden kann. Deswegen werden einige klassische Verfahren der Computerlinguistik benutzt, um die zu vergleichenden Texte zu normieren und anschließend deren Ähnlichkeit zu bestimmen.

So werden zuerst von allen Wörtern die Stämme gebildet, damit der Vergleich nicht an Flexionsformen scheitert. Danach werden Stoppwörter wie “sein” oder “und” aus dem Text entfernt, da diese für die Ähnlichkeiten von Texten zumeist nicht relevant sind. Genauso verfahren auch Suchmaschinen wie google, wenn sie Texte nach Suchbegriffen durchsuchen. Dabei werden auch nicht signifikante Wörter aus dem Text entfernt.

Normalerweise würde bei einer großen Menge von Texten einfach die Häufigkeit des Auftretens von Wörtern im Vergleich zu anderen Texten bestimmt werden. Dieses Maß bezeichnet den so genannten TFIDF-Wert (Term Frequency Inverse Document Frequency). Dieser Wert gibt an, wie wichtig ein Wort für einen Text ist. Texte, die dasselbe Wort auch enthalten, sind dazu umso ähnlicher, je häufiger dieses Wort auftritt. Da zum Vergleichen aber zumindest am Anfang der Korrektur nur zwei Texte vorhanden sind, nämlich der Ausgangstext, der unter Umständen leer sein kann, aber auch einen Text zum Vervollständigen enthalten kann und eine Musterlösung, wird statt dessen der Abstand zwischen dem zu korrigierenden Text und dem Ausgangstext sowie der Abstand zwischen dem zu korrigierenden Text und der Musterlösung bestimmt. Der Abstand zwischen dem Ausgangstext und der Musterlösung muss dabei nicht berechnet werden, da er bei der Berechnung herausgekürzt werden kann.

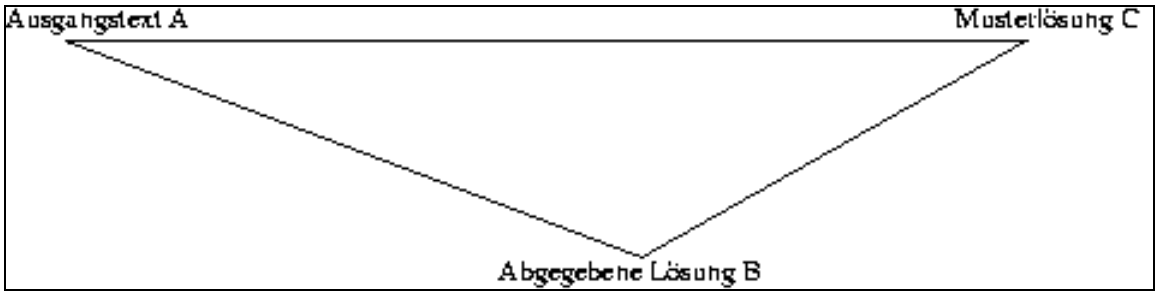

Abbildung 1:Vergleich von Texten in ViPS

Mit dieser Methode wird über die Texte jeweils ein Hashwert gebildet, der ähnliche Texte auf ähnliche Hashwerte abbildet und dann der Abstand *BC* von dem Abstand *AB* subtrahiert, um zu einer Bewertung zu kommen.

Wenn der Dozent nun bereits Texte korrigiert hat, so können auch diese, da durch die Bewertung des Dozenten ja bereits ein Abstand zur Musterlösung berechnet wurde mit in die Bewertung eingezogen werden. Durch das Einbeziehen der Bewertungen anderer Aufgaben wird die automatische Korrektur genauer und es

kann zum Beispiel einfach entdeckt werden, ob Studierende eine Lösung gemeinsam erarbeitet haben.

4.3. Prologaufgaben

Die automatische Korrektur von Programmieraufgaben anhand des Ein-Ausgabe-Verhaltens der zu bearbeitenden Programme ist verhältnismäßig leicht zu realisieren. Momentan ist dieses Vorgehen für die Programmiersprache Prolog implementiert, die Ankopplung anderer Programmiersprachen ist geplant.

Die strukturelle Analyse von Programmiercode ist bei logischen und funktionalen Programmiersprachen vergleichsweise gut erforscht und bereits in verschiedenen Systemen implementiert (vgl. [And90] und [Pey99]). Imperative und objektorientierte Programmiersprachen sind diesbezüglich problematischer, die strukturelle Analyse ist in ViPS daher nicht geplant.

Um die Last auf dem Server, auf dem Stud.IP betrieben wird, nicht durch eine große Menge an ausgeführten Programmen zu erhöhen, werden Programmieraufgaben genau so wie z.B. das Zeichen von Graphen mit Octave auf einen getrennten Rechner ausgelagert. Anfragen an den Prologinterpreter und die automatische Auswertung der Aufgaben werden mit einigen Zusatzinformationen in einem XML-String an den entsprechenden Server geschickt, wo auf der entsprechenden Webseite ein Perlskript die Anfragen entgegennimmt und je nach Anfrage die entsprechende Antwort generiert. Letztendlich wird dort der Prolog-Interpreter durch eine Reihe von Shellskripte angesteuert. Das Ergebnis ist wieder in XML eingebettet und kann so einfach vom Programmcode, der die jeweilige Aufgabe repräsentiert, interpretiert werden.

5. Neuerungen

5.1. Vorteile der Stud.IP Integration

Im Gegensatz zur Integration von ViPS in Stud.IP hätte auch eine Anbindung des Prüfungssystems an die Lernplattform vorgenommen werden können, ähnlich der Anbindung von Ilias an Stud.IP. Die Entscheidung zur Integration ist bewusst getroffen worden, da sich dadurch eine Reihe von Vorteilen bietet. So kann bei der Entwicklung von Stud.IP zum Beispiel, die Datenbankanbindung und das Sessionmanagement übernommen werden, so dass an dieser Stelle kein zusätzlicher Entwicklungsbedarf vorhanden ist. Auch das Rechtemanagement kann von Stud.IP geerbt werden. Leider gibt es noch keine einheitliche API für Stud.IP Designelemente, die für die Gestaltung der Oberfläche genutzt werden können. Nur für einzelne Teile wie Tabellen konnten diese übernommen werden.

Auch für die Anwender ergeben sich einige Vorteile. So müssen sie sich nicht extra authentifizieren oder eine neue Bedienung erlernen, da diese von Stud.IP übernommen wird. Auch gibt es einige Funktionen, wie zum Beispiel die Gruppenverwaltung, die anderswo in Stud.IP zu in gleicher Form zu finden sind und auch genau so einfach zu bedienen sind. Des Weiteren kann der Anwender sowohl beim Erstellen als auch beim Beantworten und Korrigieren die Formatierungen verwenden, die auch in Stud.IP verwendet werden kann. So können neben normaler

Textauszeichnung wie kursiv oder fett, Hoch- und Tiefstellung des Textes auch Grafiken eingebunden, Links gesetzt oder mit Hilfe von LaTeX mathematische Formeln eingegeben werden.

5.2 ViPS ohne Stud.IP

Obwohl das Haupteinsatzgebiet von ViPS sicherlich innerhalb von Stud.IP liegt, da so viele Vorteile von Stud.IP mitgenutzt werden können, ist es auch möglich, den Aufgabenteil von ViPS ohne Stud.IP zu benutzen.

Dazu wurde das System so angelegt, dass die Aufgabentypen unabhängig von Speicherort (Datenbank, Dateisystem oder temporärer Speicher), Gruppenmodus (Bearbeitung einzeln oder in der Gruppe) und der verwendeten Auswertungsstrategie. Intern wird zur Repräsentation der Lösungen und der Aufgaben eine XML-Struktur verwendet. Jedoch werden in den Aufgaben einige Funktionen benutzt, die in Stud.IP enthalten sind, wie zum Beispiel die Formatierung des Textes. Diese müssen bei Verwendung der Aufgaben innerhalb eines anderen Kontextes reimplementiert werden.

Da das System nach leichter Modifikation ohne Stud.IP benutzbar ist, wurde es erfolgreich in pmwiki (http://www.pmwiki.org) integriert. Aufgaben werden dabei mit einer speziellen Syntax in pmwiki eingegeben und können dann als Selbsttest innerhalb einer Seite benutzt werden. Geplant ist des Weiteren, dass Lösungen der Aufgaben innerhalb von pmwiki gespeichert werden.

Darüber hinaus wird ViPS seit einiger Zeit auch zur Darstellung des Selbstlerncontent in ImpulsEC (http://www.impuls-ec.de/) eingesetzt. Ein Besonderheit ist dadurch gegeben, ImplusEC Stud.IP zwar zur Kursverwaltung, nicht aber zur Darstellung des Contents nutzt. Das ViPS kann daher nur zur Erstellung der Aufgaben genutzt werden, die Darstellung und Auswertung der Aufgaben findet unabhängig vom ViPS statt. Die Aufgaben werden in diesem Fall in einer separaten Datenbank verwaltet.

6. Kursbewertung im ViPS: Punkteverteilung und Notenberechnung

Neben der rechnergestützten Abwicklung von Hausaufgaben und Klausuren sowie deren zum Teil selbsttätig ablaufender Korrektur bietet das ViPS die Funktion der automatischen Evaluierung der Ergebnisse gelöster Aufgabenblätter. Dabei beschränkt sich die Anwendbarkeit nicht auf einzelne Klausuren, sondern die gesamte Kursauswertung inklusive der Berechnung der Endnoten wird automatisch vorgenommen.

Der Dozent hat vielfältige Möglichkeiten, den Berechnungsablauf auf das spezifische Niveau der einzelnen Aufgabenblätter abzustimmen. Jede Klausur und jeder Aufgabenblock wird mit einer eigenen prozentualen Gewichtung versehen, die die verhältnismäßige Wichtigkeit oder Komplexität der Aufgabenblätter zueinander ausdrückt. Außerdem kann der Dozent für jedes Blatt eine individuelle Skalierung festlegen: Durch Angabe eines Mindestprozentwertes zum Bestehen einer Aufgabe sowie eines zum Erreichen der Bestnote nötigen Wertes können die

Bewertungskriterien eines einzelnen Aufgabenblockes somit entweder verschärft oder gelockert werden.

Die Bewertung beginnt mit der Betrachtung eines einzelnen Aufgabenblockes bzw. einer Klausur. Anhand der korrigierten Lösungen werden die vom Studenten erreichten Punkte p_{abs} ermittelt und deren Verhältnis zur maximal erreichbaren Punktzahl p_{max} berechnet. Die so errechnete Verhältnispunktzahl

$$p_{rel} = p_{abs} : p_{max}$$

bewegt sich im Intervall [0; 1], kann aber auch – falls Bonuspunkte verteilt wurden – größer als 1 sein.

Um aus der Verhältnispunktzahl p_{rel} die skalierte Punktzahl p_{skal} zu errechnen, werden die vom Dozenten angegeben Werte b_{min}, b_{max} □ [0; 1] zum Bestehen der Aufgabe, respektive zum Erreichen der Bestnote, benötigt. Der Notenbereich [b_{min}; b_{max}] wird durch Subtraktion von b_{min} auf [0; $b_{max} – b_{min}$] verschoben und durch Multiplikation mit dem Skalierungsfaktor $f_{skal} = 1 : (b_{max} – b_{min})$ auf [0; 1] gestreckt. Die selben Transformationen auf p_{rel} angewandt ergeben

$$p_{skal} = (p_{rel} – b_{min}) \cdot (1 : (b_{max} – b_{min})).$$

An diesem Wert p_{skal} läßt sich die Leistung des Studenten leicht ablesen: Bei einem negativen p_{skal} (wegen $p_{rel} < b_{min}$) wurde das Aufgabenblatt nicht bestanden, ein p_{skal} □ [0; 1] (wenn $b_{min} \leq p_{rel} \leq b_{max}$) entspricht einer Note innerhalb des „Notenbereichs“ vom Bestehen bis zur Bestnote, während ein $p_{skal} > 1$ (für $p_{rel} > b_{max}$) bedeutet, daß der Student mehr Punkte erreicht hat, als für die Bestnote nötig gewesen wären. Dieser Umstand soll jedoch in der weiteren Auswertung und Notengebung nicht berücksichtigt werden, weshalb größere Werte automatisch auf 1 zurückgesetzt werden.

Die Gewichtung der einzelnen Aufgabenblöcke wird vorgenommen, indem die skalierte Punktzahl p_{skal} mit dem durch den Dozenten zugewiesenen Gewichtungsfaktor f_{gew} □ [0; 1] multipliziert wird und so die gewichtete Punktzahl

$$p_{gew} = p_{skal} \cdot f_{gef}$$

ergibt.

Auf diese Weise wird die gewichtete Punktzahl p_{gew} eines jeden vom Studenten bearbeiteten Aufgabenblocks berechnet und zu einer Gesamtpunktzahl $\Sigma(p_{gew})$ aufsummiert. Gleichzeitig werden die Gewichtungsfaktoren f_{gew} aller bestandenen Aufgabenblöcke – was anhand der jeweiligen skalierten Punktzahlen p_{skal} überprüft wird – zu einer Summe $\Sigma(f_{gew})$ addiert; die f_{gew} der nicht bestandenen Blöcke erscheinen nicht in der Summe. Ferner werden im folgenden die vom Dozenten gesetzten Schwellenwerte zum Erreichen der einzelnen Notenstufen $b_{4,0}$, $b_{3,7}$, …, $b_{1,0}$ zur Berechnung der Endnoten benötigt.

Die Schwellenwerte werden – analog zu den Verhältnispunktzahlen p_{rel} – skaliert durch Anwendung der selben oben genannten Formel, so daß $b_{4,0} = 0$ und $b_{1,0} = 1$ und sich die übrigen Werte proportional zu den ursprünglichen – nicht skalierten – Abständen zueinander anordnen. Die so erhaltenen skalierten Notengrenzen lassen sich dadurch, daß hier die selben Transformationen durchgeführt wurden, direkt mit der skalierten, gewichteten und über alle Blöcke aufsummierten Punktzahl $\Sigma(p_{gew})$ des Studenten vergleichen, wodurch sich durch einfaches „Ablesen“ die Endnote

entscheiden läßt.

Ungeachtet des so errechneten Resultats erhält ein Student in jedem Fall die Note 5,0, wenn $\Sigma(f_{gew}) < b_{4,0}$, also wenn sich die Gewichtungen der bestandenen Blöcke nicht mindestens zur Grenze zur Note 4,0 summieren, weil nicht genügend Aufgabenblöcke bzw. nur solche mit zu geringer Gewichtung gelöst werden konnten. Dabei ist es nicht von Bedeutung, wie erfolgreich der Student bei der Lösung der bestandenen Aufgaben war – wenn nicht genügend Blöcke erfolgreich gelöst wurden, gilt der Kurs definitiv als nicht bestanden.

7. Fazit

ViPS und seine Vorgänger haben sich in den letzten Jahren sowohl als Teil von Stud.IP als auch im eigenständigen Betrieb als robust und zuverlässig erwiesen. Der aktuelle Schwerpunkt der Entwicklung liegt daher auf der Erweiterung durch neue Features und auf der Verweinfachung der Bedienung des Systems. So wird derzeit eine elaboriertere Aufgabenverwaltung entwickelt. Darüber hinaus werden in Zusammenarbeit mit der Universität Oldenburg eine Peer-Reviewing-Erweiterung für ViPS entwickelt und eine Reihe mathemathischer Aufgabentypen integriert.

ViPS ist Open-Source-Software. Es ist daher frei verfügbar und kann durch universitätseigenes Personal an spezifische Gegebenheiten und Anforderungen angepasst werden. Vor allem für Standorte, an denen bereits Stud.IP betrieben wird, bietet sich der Einsatz von ViPS an.

8. Literatur

[And90] Anderson, J., Boyle, C., Corbett, T. & Lewis, M.: Cognitive Modeling and Intelligent Tutoring. In Clancey J. & Soloway, E. (Hrsg.). MIT-Press, Cambridge, London, 1990. S. 7 - 51

[Gus03] Gust, H., Hügelmeyer, P., Mertens, R. & Rollinger, C.: Automatische Abwicklung von Klausuren mit dem MVC. In: V. Dötsch, G. Schade & K. Hering (Hrsg.), e-Learning and beyond. Proceedings of the Workshop on e-Learning 2003, HTWK Leipzig, 14.-15. Juli 2003. Leipzig: Hochschule für Technik, Wirtschaft und Kultur (FH), Fachbereich Informatik, Mathematik und Naturwissenschaften. S. 157-165

[HM04] Hügelmeyer, P. & Mertens, R.: Virtuelles Prüfungssystem. In Hamborg, K.-C., Knaden, A. (Hrsg.), Good Practice: netzbasiertes Lehren und Lernen an Universitäten - Erfahrungen mit verschiedenen Einsatzszenarien von e-Learning an der Universität Osnabrück. epos Media, Osnabrück 2004. S. 105-117

[Knu98] Knuth, D.: The Art of Computer Programing Volume 3: Sorting and Searching, Second Edition. 1998, Eddison Wesley

[Pey99] Peylo, C., Teiken, W., Rollinger, C. & Gust, H.: Der VCProlog-Tutor, eine Internet-basierte Lernumgebung. Künstliche Intelligenz (4/1999), 32-36.

[Pey00] Peylo, C., Thelen, T., Rollinger, C. & Gust, H.: A web-based intelligent educational system for PROLOG. In C. Peylo (Hrsg.), Proceedings of the International Workshop on Adaptive and Intelligent Web-Based Education Systems held in conjunction with ITS 2000.

[St05] Stud.IP Developer Core Group: Die Stud.IP-Philosophie. http://www.studip.de (24.05.2005).

[The04] Tobias Thelen: Elektronische Anmeldeverfahren für Präsenzlehrveranstaltungen. In: Knaden, Andreas; Hamborg, Kai-Christoph (Hrsg.): Good Practice: Netzbasiertes Lehren und Lernen. Osnabrücker Beiträge zum medienbasierten Lernen, Band 1. Electronic Publishing Osnabrück, 2004. S. 5-18

[TR03] Thelen, T.; Rieker, D: WebCT an der Universität Osnabrück: Erfahrungen und Konsequenzen, in: Tagungsband zum Workshop on e-Learning 2003, Leipzig 2003, S. 139-148.

[Wag00] Wagner, E. (Hrsg.): Projektverbund »VIRTUELLER CAMPUS« der Universitäten Hannover - Hildesheim - Osnabrück (Projekt VC I: 1997 - 2000) Abschlussbericht. http://www.uni-hildesheim.de/ZFW/vc/vc-bericht.pdf (24.6.2005).

[WB04] Wollermann, T. & Busch, K.: Einsatz von WikiWikiWebs zur Projektkoordination und Projektarbeit in Seminaren. In Hamborg, K.-C., Knaden, A. (Hrsg.), Good Practice: netzbasiertes Lehren und Lernen an Universitäten - Erfahrungen mit verschiedenen Einsatzszenarien von e-Learning an der Universität Osnabrück. epos Media, Osnabrück, 2004. S. 39-52